ANALYSIS AND SHORT-TERM PREDICTION OF URBAN TRAFFIC IN MADRID UNDER THE INFLUENCE OF MULTITUDINARY EVENTS USING MACHINE LEARNING TECHNIQUES

SERGIO PINA LAGUNAS

MÁSTER EN INTERNET DE LAS COSAS. FACULTAD DE INFORMÁTICA.
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin de Máster en Internet de las Cosas

Septiembre 2018

Directores:
Rubén Fuentes Fernández
Jorge J. Gómez Sanz

Calificación: 8

Autorización de Difusión

SERGIO PINA LAGUNAS

3 de Septiembre de 2018

El/la abajo firmante, matriculado/a en el Máster en Internet de las Cosas de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "Analysis and prediction of urban traffic in Madrid under the influence of multitudinary events using Machine Learning techniques", realizado durante el curso académico 2017-2018 bajo la dirección de Rubén Fuentes Fernández en el Departamento de Ingeniería del Software e Inteligencia Artificial, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Resumen en castellano

En este trabajo se busca analizar a partir de los datos de tráfico de la ciudad de Madrid, y de ciertos eventos que suceden en la ciudad, la aparición de situaciones de congestión de tráfico. El objetivo es evaluar si estos eventos tienen influencia negativa en el tráfico urbano y en qué puntos de la ciudad. En la medida de lo posible, se busca también alertar de forma anticipada acerca de esas congestiones.

El anterior análisis se ha organizado en varias partes. Primero se presenta un análisis de la información recabada. En él, se pretende observar cuándo y en qué puntos de las vías urbanas ciertos eventos con un desplazamiento masivo de gente resultan más influyentes negativamente en el desarrollo del transporte. A partir de esta información pasada, se realiza una propuesta para un sistema de aprendizaje y clasificación que sea capaz de utilizarla para predecir estos problemas en situaciones similares con algunas horas de anticipación.

Respecto a los datos recabados, han sido datos históricos de tráfico y eventos en la ciudad. Los datos de tráfico recogen métricas, (ej. la intensidad de vehículos o la carga de la vía en un intervalo de tiempo) en diferentes puntos geográficos. A fin de identificar eventos relevantes, también se han recopilado para las fechas consideradas información sobre grandes eventos de masas (ej. partidos de fútbol o periodos de compras). Estos datos se han preprocesado, incluyendo tareas de homogenización de las representaciones, y limpieza de valores perdidos o incompletos. También se ha cruzado información de forma que fuera coherente con el análisis que se pretendía realizar. Tras ello se han aplicado algoritmos de clasificación y aprendizaje (ej. árboles de decisión o k-vecinos más cercanos) y se ha desarrollado el modelo previamente mencionado.

Finalmente, se han expuesto los resultados y conclusiones obtenidos del proceso. En ellos se exploran los resultados obtenidos por los algoritmos clasificadores , se explica cómo afectan los diferentes eventos a los puntos geográficos en términos de las mediciones que se observan en ellos, y se sugieren formas de mejorar los modelos en futuros trabajos.

Este trabajo ha sido realizado con el apoyo de los siguientes proyectos: “Diseño Colaborativo para la promoción del bienestar en Ciudades Inteligentes Inclusivas' (TIN2017-88327-R) financiado por el Ministerio para la Economía y la Competitividad de España; y MOSI-AGIL-CM (S2013/ICE-3019) co-financiado por la Comunidad de Madrid y los fondos estructurales de la Unión Europea FSE, y FEDER.

Palabras clave

Tráfico Urbano, R, Aprendizaje Automático, Congestión vial

Abstract

This work aims at analyzing the appearance of traffic congestions in the city of Madrid. The analysis uses data on traffic and certain events in the city. In this context, the goal is twofold. First, to study which of these events have a real influence in the generation of these congestions and where in the city. Second, being able to alert with some anticipation about these situations.

The previous analysis has been organized in several parts. First, an analysis of the considered information is introduced. Its goal is to observe when and in which points of the urban roads certain events that imply a massive displacement of people have a more negative impact on traffic. Then, from this information, a proposal of a learning and classification system able to use that information to predict these problems in similar situations is made.

Regarding data, the work gathered historic data on traffic and events. Traffic data include metrics (e.g. intensity of vehicles or road load in a time interval) in different geographical points. Information on events includes football matches of the main city teams and the Christmas shopping period. These data have been preprocessed, including tasks of homogenization of representations, and cleaning of lost or incomplete values. Information has also been merged to make it coherent with the intended analysis techniques. After that, the work applied several classification algorithms (e.g. decision trees and k-nearest neighbors) and developed the prediction model previously mentioned.

Finally, results and conclusions have been discussed. This explores the results the classifier algorithms obtained, explains how the different events affect to the geographic points in terms of the observed traffic data, and suggests ways to improve them.

This work was supported by the following projects: “Collaborative Design for the Promotion of Well-Being in Inclusive Smart Cities'” (TIN2017-88327-R) funded by the Spanish Ministry for Economy and Competitiveness; and MOSI-AGIL-CM (S2013/ICE-3019) co-funded by Madrid Government, EU Structural Funds FSE, and FEDER.

Keywords

Urban Traffic, R, Machine Learning, Road Congestion

Index of contents

Index of figures

Index of tables

Index of acronyms

AEMET: Agencia Estatal de METereología

A.I.: Artificial Intelligence

ARIMA: Autoregressive Integrated Moving Average

ATM: Club Atlético de Madrid

CDT: Cell Dwell Time

CSV: Comma-Separated Values

e.g.: exempli gratia (for example)

GUI: Graphical User Interfaces

IDE: Integrated Development Environments

i.e.: id est (that is)

IoT: Internet of Things

ITS: Intelligent Transportation Systems

K-NN: K-nearest neighbors

PC: Plaza Callao

RM: Real Madrid Club de Fútbol

SB: Santiago Bernabéu

TLRN: Time-Lag Recurrent Network

UTM: Universal Transverse Mercator

WM: Wanda Metropolitano

Chapter 1 - Introduction

Urban traffic is a growing problem in big cities, especially in the ones whose public transport or alternative ways of moving within the town are not fully developed. These problems are exacerbated when an event gathers crowds around a concrete geographical point. The Internet of Things (IoT) may play an important role regarding solving or mitigating this problem.

1.1 Motivation

The growth of world population and the migration from smaller to bigger cities is causing the appearance of huge urban areas. As cities grow bigger, so their traffic problems do in relevance and frequency. These problems have a great impact in the life of city inhabitants. They are a source of pollution, both air and acoustic, that negatively affects people' health. For instance, Jerret and Sears (2004) pointed out that continued exposure to air pollution is related to higher mortality rates. Moreover, congestions also have economic effects. For instance, Weisbrod et al. (2003), used statistical models to show the negative impact of traffic congestions on the productivity of certain businesses.

At the same time, cities are becoming early adopters of Internet of Things (IoT) solutions to improve the quality of life of their inhabitants under the concept of *Smart Cities*. The prices of the devices needed to deploy big sensor networks and process their data are becoming more affordable, and communication technologies are widely deployed. As this happens, IoT solutions for Smart Cities grow, implementing services over those data and infrastructures that add value to the people living in them. Over the last years, several cities have become testbeds for the Smart City concept. Zanella et al. (2014) provided a comprehensive survey of the technologies, protocols and architecture for urban IoT solutions, and a discussion about the example of the city of Padova (Italy).

In this context, IoT solutions to deal with traffic congestion are being explored. This work belongs to that line.

This work wants to acknowledge the support of the following projects: “Collaborative Design for the Promotion of Well-Being in Inclusive Smart Cities”” (TIN2017-88327-R) funded by Spanish Ministry for Economy and Competitiveness; and MOSI-AGIL-CM (S2013/ICE-3019) co-funded by Madrid Government, EU Structural Funds FSE, and FEDER.

1.2 Main objectives

The main goal of this work is being able to identify and predict in the short-term traffic congestions using historical data on traffic and massive events in the city of Madrid, Spain. To do that, it studies when and where traffic can grow to a critical point around those events using classification algorithms. This analysis is the basis to build a model that learns from the information of past events to predict congestions.

The data considered so far include historic ones on traffic and events. Traffic data include geographic and numeric congestion-related information. These have been linked to several additional datasets to identify the events having a negative impact of traffic. These datasets are related to meteorological conditions and massive events. These integrated data support the identification of the points in the city where it is more common to find congestion problems when certain events are happening.

The processing of the data and the recognition of congestions proposed in this work, in the context of an IoT solution, could be used as inputs for a system able to communicate these situations and to propose alternative routes or further recommendations. This can support decisions on transport in different contexts. It may provide useful information regarding planification of public transport routes and frequencies. Besides, people who need to move in the city before, during or after those events can use it to plan its transport.

1.3 Organization of this work and structure of the document

In order to accomplish the objectives exposed in section 1.2, the work has been organized as follows:

- *Data acquisition from sources in the Internet*. Data were acquired from the open data webpage of the local government of the municipality of Madrid and the webpage of the Spanish Weather Agency (*Agencia Estatal de METeorología,* AEMET). The data on events in the city (e.g. football matches) was manually prepared for this work.
- *Data cleaning and shaping*. Raw data needed to be prepared for the analyses. The complete original datasets were too detailed for the current work (for instance regarding sampling time). This has the additional problem of high needs of computational resources. Moreover, as they were obtained from different sources, there were inconsistencies in formats and representations, missed values, and errors in measures.
- *Study of relationships among variables*. The work started with the study of the potential relationships among traffic parameters in a point and the sequence of events, looking to identify conflictive. This was done with the K-means algorithm for clustering. This provided an idea about the type of congestions that could happen and the differences between the kind of roads where the traffic measure points are located. It also helped to stablish the threshold values of parameters to consider that there is a traffic congestion issue in a given point and time interval.
- *Congestion prediction using the historical data*. The work also studied the relationship between the state of the traffic in two consecutive time intervals as a classification problem using decision trees and k-nearest neighbors (k-nn). Most works approach prediction using statistical methods or neural networks, but this work decided to use the classification approach to evaluate its effectiveness and learn about its advantages/disadvantages. Thus, prediction is regarded here as a classification problem. These algorithms were chosen because there are very few approaches to short-term traffic-congestion predictions that have considered them. Several experiments were carried out, including training and evaluation of their performances.

The structure of the rest of the document closely follows these steps. Chapter 2 discusses the state of the art and introduces the tools and techniques used to analyze the data and build the

model. Chapter 3 presents the data sources and the data preprocessing for the next steps. Then, the different algorithms are applied. Chapter 4 reports the study with clustering algorithms, and . Chapter 5 with classification algorithms. The results are discussed globally in Chapter 6. Finally, Chapter 7 includes conclusions and future work.

Chapter 2 - State of the art

In this Chapter, the current state of the art related to the work is discussed. First, it reviews works on Smart Cites. Then, it approaches the analysis of urban traffic congestion. Finally, it presents the techniques and tools used in this work.

2.1 Smart City services for traffic management

Smart City services have already been considered for traffic management. This includes the detection and mitigation of congestion. This section focuses on some of the system that have approached this functionality.

Most of works on detection rely on techniques of Computer Vision, given that they are quite developed regarding the monitoring of places with cameras. For instance, Buch, Velastin and Orwell (2011) analyzed urban traffic. Their work considered that, under decreasing costs of hardware and software related to traffic surveillance (e.g. cameras and computers), it was feasible extending semi-automated monitoring of traffic. These authors not only considered traffic congestion, but also traffic rule violations or the interactions between vehicles regarding detection and classification issues.

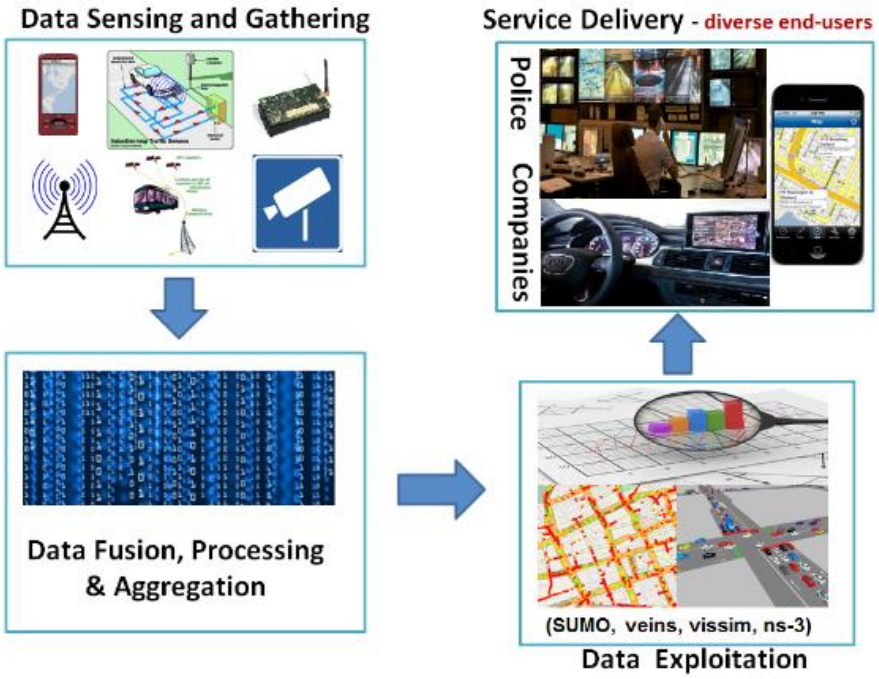
There are also works that go beyond and consider traffic management. For instance, France and Ghorbani (2003) propose a hierarchical multi-agent system to manage efficiently metropolitan traffic. They developed patterns for traffic lights to reduce congestion in two different scenarios: traffic accidents and the moments of the day closer to the starting of working hours. These patterns redirected traffic from the congested areas, making it more fluent and allowing it to get back to standard levels. It mainly works reducing the congestion in critical points by redirecting it to secondary ways.

These works can usually been improved when considering communication aspects with participants to improve the interactions. For instance, Djahel et al. (2015) considered in their

review the use of smart vehicles and social media to deal with congestions caused by events such as accidents or the initial and final working hours.

The previous review (Djahel et al., 2015) also discusses some design principles for Traffic Management Services, in particular the four-phases data life cycle as shown in Figure 2.1. The current work has been focused mainly on the three first stages.

Figure 2.1 Data life cycle (Djahel et al., 2015)



2.2 Analysis of urban traffic congestions

This section focuses on works about the analysis and prediction of traffic congestion. It considers the techniques and algorithms used in this domain.

The location and intensity of congestions depends on multiple factors: the city map with the design of the street networks; kinds of roads and streets; the location of the points of interest; and even the time of the year. The first step is to have suitable data to study the problem. In this

problem, studies frequently resort to government data. For IoT solution, data obtained from sensors are used more, for instance from cameras or mobile devices.

An example of the use of mobile devices in (Hongsakham et al., 2018). They used data about the Cell Dwell Time (CDT) from cellular networks to identify congestions. It classified measures within three degrees of congestion. Those results were compared with human opinions, showing the good performance of the approach.

In the case of this work, the Madrid's government open data web page is a key source of data. Its detection systems consist mostly of electromagnetic ties placed under the road which detect the metallic mass of vehicles passing over them. They have the limitation of restricting data acquisition to one point instead of the whole road. Besides, they do not have any support from other systems (e.g. camera images) to give meaning to the acquired data.

A second aspect has been the identification of congestion problems. Researchers have used different techniques over different datasets.

Clustering algorithms have been used to partition data. In (Aslam et al., 2012), they identified conflictive geographic areas in terms of traffic congestion. The work developed a model that used traffic data collected by a roving sensor network of taxi probes to analyze patterns or infer them in real time. The k-means algorithm was applied to partition the area of Singapore in regions with different traffic patterns. In (Spieser et al., 2014), the same algorithm was used to identify patterns for an automated mobility-on-demand system. K-means allowed partitioned Singapore in 100 regions using the data gathered in a survey to households about characteristics of their daily trips to and from work The analysis showed that a change into a shared-vehicle mobility solution would reduce the number of vehicles to 1/3 of the current number, and suggested an automated system for its management.

There are also works aimed at prediction. Again, several techniques have been applied.

Decision trees have been considered to perform short-term predictions related to traffic issues. For instance, accident prediction with decision trees appears in (Zheng et al., 2016). The authors justified their use against other algorithms (e.g. regression models) with their versatility to deal with large data sets, missing values, and the lack of predefined relationships between target variables and predictors. Similar reasons appear in (Chang and Wang, 2016), where decision trees are used to classify the severity of traffic injuries. They built a model using accidents data and classification and regression trees to establish a relationship between the severity of driver's injuries and characteristics of the vehicles, environment and accidents. Finally, and closer to our work, decision trees have been used to predict the duration of traffic incidents in freeways (Liu et al., 2015).

The k-nn technique has also been used for predictions. For instance, (Zhang et al., 2013). developed a model based on this technique to build a model for short term prediction using a historical database of traffic parameters from Shanghai's expressway.

Techniques based on the temporal dimension of data are also suitable in this context.

For instance, (Statopoulos and Karlaftis, 2003) developed multivariate time-series state space models for traffic modeling and prediction in Athens. Their premise is that successful Intelligent Transportation Systems (ITS) to work with urban congestion depend on the provision of accurate real-time information about travel volumes and speeds, occupancies... They compared several models and found that different models worked better for different moments of the day and the prediction accuracy was better for the multivariate ones than for the univariate ones.

Other tools used for traffic flow prediction are seasonal time series models. In (Williams et al., 1998), seasonal Autoregressive Integrated Moving Average (ARIMA) and Winters exponential smoothing models were developed to do forecasts in two main roads in Beltway, North Virginia, U.S.A. The work found that ARIMA got better results for this concrete problem than k-nn, neural network and historical average models. ARIMA models consist on linear estimators regressed on past values of the modeled time series for autoregressive terms and past prediction

errors for moving average terms. Exponential smoothing models consist on linear estimators that assign more weight to the recent values while in the past ones decrease exponentially.

Considering that any vehicle can act as a node providing useful information for traffic modelling, a different approach used large scale taxi GPS traces to do it (Castro et al., 2012). The authors here used the data provided from 5000 taxis equipped with GPS. These taxis created logs with the routes followed measuring the density on the road and presented a method that determined the capacity of the road.

Neural networks were also considered. In (Dia, 2001), an object-oriented neural network model is built to predict short-term traffic conditions in a highway in Australia. Concretely, it uses a Time-Lag Recurrent Network (TLRN) with good results. These networks add short-term memory structures to the standard neural networks, which make them more suitable for this kind of problem.

There are also works that addressed comparisons between several approaches for prediction.

In (van Hinsbergen et al., 2007) several machine learning algorithms were tested, including the decision trees used in this work. The results suggest that there is no solution suitable for all the situations, and the best approach could be to combine them.

In (Lippi et al., 2013), time-series analysis and supervised learning were compared for the short-term prediction of traffic congestions. The work reviews the performance of several statistical and machine learning methods for this problem. Different versions of ARIMA are tested. Artificial neural networks, support vector regression (with radial basis function, seasonal mean and random walk), Bayesian networks, echo-state networks, nonparametric regression and approaches based on the genetic algorithm are considered regarding machine learning. The results showed that seasonality must be considered to achieve a considerable accuracy. Also, that there are high computational needs for both training and prediction, as relevant volumes of data must be

considered. Seasonal ARIMA (SARIMA) with a Kalman filter was the best model on average in this work in terms of performance, accuracy and cost during highly congested periods.

In the current work, clustering and classification algorithms have been considered for the short-term prediction of traffic congestions. The reviewed works support the use of these two classifiers (i.e. decision tress and k-nn) to make the predictions proposed in Chapter 1. Both have been used in traffic studies with similar purposes, though this specific problem is a quite novel domain that deserves attention.

Here, the goal of clustering is detecting common patterns in measures, observe the influence of the events selected, and establish threshold values for the parameters related with congestion, in order to determine where and when might exist a traffic problem. **When these parameters meet the conditions, our work considers that there is a congestion (here called an *activation*)**. The goal for classification is predicting these activations for the next time interval.

As sections 3.1 and 5.1 explain, prediction of congestion in this work consists in determining if a variable has a discrete value of 0 (there is no congestion issue) or a discrete value of 1 (a congestion issue exists), i.e. a binary classification problem. This, added to the problems to work with big datasets and regression compared to classification (Zheng et al., 2016), lead here to choose this last option. Though some techniques, like Support Vector Machines (SVM) can be used in both context, there are some differences. In the case of SVM (Yu and Kim, 2012), they were developed for classification and, afterwards, extended to regression and preference learning.

2.3 Machine Learning Techniques

Engineers and scientists have always pursued the goal of computers able to learn (Kubat, 2017). It is the age of Machine Learning. Its roots can be traced back to works on statistics or the first researches with computers in the 1950s. There are multiple reviews and books about the topic, like the book *Machine Learning: The AI Approach* (Mitchell et al., 2013) with its first edition in 1983. This work borrows a bit of that knowledge to carry out the analysis proposed and build a

working predictive model. Next sections review the main concepts and techniques from Machine Learning used here.

2.3.1 Unsupervised Learning: Clustering algorithms

Unsupervised learning is characterized by the fact that it only has input data, but there are not classified of expected output data a priori. For instance, clustering methods or some association rules belong to this learning. In the case of classification, it does not use any classification label to perform its work, i.e. the examples are not previously classified as it happens in supervised learning. Hence, clustering algorithms belong to this kind of learning.

The main goal of clustering algorithms is to discover useful information and properties in the available data, grouping the provided examples by their characteristics in entities called clusters. Ideally, these clusters should not overlap, so each example should belong to only one cluster. Besides, within a cluster, the distance among its examples should be minimal. If these constraints are met, the points of information are well differenced, and the different kinds of examples (i.e. the clusters) can be determined.

In clustering, given a set of examples, it must be determined to which cluster each of them belongs. It is assumed that each example is defined by an attribute vector x and each cluster by its centroid, which is described as an attribute vector too. Suppose there are N clusters whose centroids are denoted by ci, where i is placed between 1 and N. The example x is separated from each centroid a certain distance $d(x,ci)$. If $d(x,cp)$ is the smallest of these distances, x should be assigned to the cluster cp.

Clustering algorithms have three main advantages. First, they are good to estimate missing attribute values. Frequently, data sources are not totally robust. Sometimes measures are not taken correctly or not taken at all. Algorithms usually need complete datasets to work properly, so, sometimes these missing values need to be estimated. For example, a value may be estimated taking the average for that concrete characteristic in the cluster assigned. In second place, cluster centroids are a good start point to classifier algorithms that operate with centers, such as Radial

Basic Function Networks or Bayesian learners (Kubat, 2017). Finally, outcomes from clustering algorithms may be useful for supervised learning. Clusters may work as groups of examples of a class, so developers can identify them first and then label them properly.

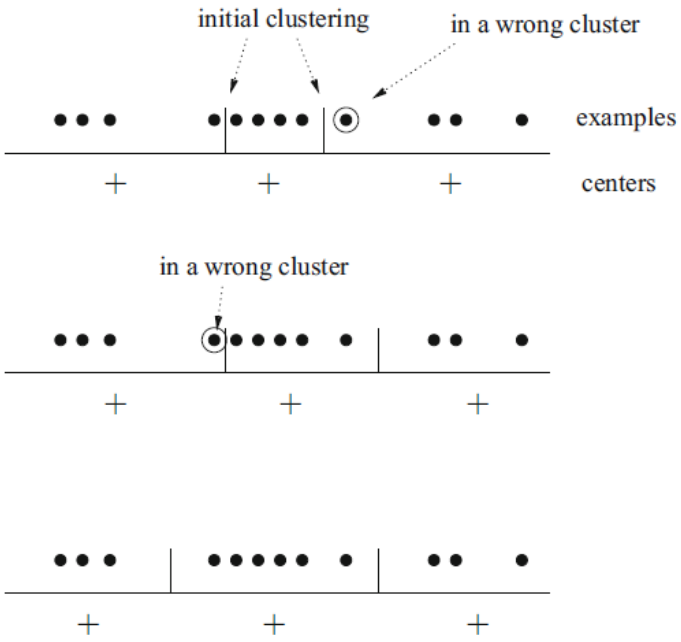
2.3.1.1 K-means

The simplest and most well-known clustering algorithm is K-means, whose *K* stands for the number of clusters in which the algorithm divides the set of examples. This number is supplied by the user (Kubat, 2017).

Given a number of examples, the algorithm works as follows. In the first step, it randomly create *K* clusters and assign examples to them. Next, it calculated the coordinates of the centroids. Then, the algorithm calculates the distance of each example to each centroid, looking for examples assigned to a wrong cluster. Those examples wrongly assigned, are reassigned to their closest cluster. If there are changes from the previous allocation to clusters, the centroids of the affected clusters need to be recalculated. The algorithm repeats this process as many times as needed until every example is assigned to a cluster and never reassigned, which means that clusters do not overlap. This process is shown graphically in Figure 2.1 considering one point in each iteration (extracted from Kubat, 2017).

The points are the examples, the crosses the centroids and the spaces between bars the clusters defined. The set consists of thirteen examples, defined by a single numeric attribute, distributed in three clusters.

Figure 2.2: k-means procedure



2.3.2 Supervised Learning

Supervised Learning, unlike unsupervised, maps inputs to outputs based on labeled training data previously provided. There are many algorithms with this approach. Below, the ones considered in this work are described.

2.3.2.1 Decision trees

Decision trees fall under the description of classifier algorithms. In other classifier algorithms, all attributes are evaluated at the same time before giving the output. However, decision trees work with attributes one at a time depending on the situation (Kubat 2017).

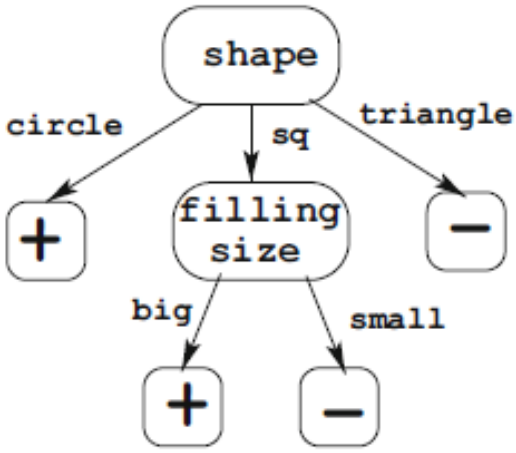
A decision tree consists of a root, nodes and leaves. The root is the topmost node, from which attributes are evaluated and a decision for the final value is taken. An example to be classified is evaluated in the root, then goes through the nodes that satisfy the conditions, until it reaches one of the leaves, which are the final nodes of the trees. The example is labeled with the class of the leaf where it has arrived.

Table 2.1 and Figure 2.1 show an example extracted from Kubat, 2017. Table 2.1 has eight training examples for classification. They have three attributes and are classified as positive or negative. Figure 2.1 shows an example of decision tree for the given training set.

Table 2.1: Set of examples for decision tree

Example	Crust Size	Shape	Filling Size	**Class**
E1	Big	Circle	Small	**Pos**
E2	Small	Circle	Small	**Pos**
E3	Big	Square	Small	**Neg**
E4	Big	Triangle	Small	**Neg**
E5	Big	Square	Big	**Pos**
E6	Small	Square	Small	**Neg**
E7	Small	Square	Big	**Pos**
E8	Big	Circle	Big	**Pos**

Figure 2.3: Decision tree for the given set



In the root, the shape is evaluated. If it is a circle, the example is classified as positive, negative in case of triangle. However, if the shape is square, the filling size is evaluated, classifying the example as positive if it is big, and negative in case of small.

In this work, results for decision trees are evaluated using the concepts of accuracy and precision. Accuracy is the ratio of good predictions, i.e. the number of positive instances correctly

predicted plus the number of negatives correctly predicted over the total number of predictions. Precision is the ratio of positives predicted correctly, i.e. the number of positives predicted that are true positives over the total number of positives in the data to predict.

An advantage of decision trees as opposed to other classifier algorithms is that users can intuitively understand their classifications. They offer explanations for the classifications of the examples, like "E4 is classified negative because its shape is a triangle". It is not possible to extract directly information of this kind from other algorithms such as k-nn. The rules used to make the decisions are understandable. An expert in the domain may even improve the performance of the tree adding rules and thus new nodes and branches.

2.3.2.2 k-Nearest Neighbors

This algorithm (K-NN) explores similarities between examples to determine the class of the one to classify (Kubat, 2017). To do that, it compares their attribute vectors.

When working in a discrete domain, the similarities between two examples can be computed as the number of attributes in which they differ. In case of working in a continuous domain, each example can be represented in an n-dimensional space, so a distance between pairs of examples can be calculated. Thus, the closest example is the most similar to the one that is being evaluated, the *nearest neighbor*. In the simplest version (k = 1), the class assigned to the example to classify is that of that neighbor.

Table 2.2 shows a set of examples to illustrate the explanation above in a discrete domain with k 1. The example x is the one being evaluated. The column *#differences* shows the number of differences in the attributes of x and the rest of examples. In this case, the lowest difference value is found for the example E5. Thus the class for x is the same as for E5, positive since is the training example most similar to x.

As the number of dimensions grow, the level of noise in the domain also grows. In this case, the nearest neighbor might be not fully trusted. It becomes necessary to go further and explore

a bigger number of nearest neighbors, making them vote to decide the class of the example. So the algorithm turns into the *k-Nearest Neighbors* being *k* the number of voting neighbors, which is specified by the developer. Depending on the number of dimensions and the choice of *k*, differences and ties may appear when deciding the class for a given example. The developer needs to build a mechanism to decide the class assigned in these cases (Kubat, 2017).

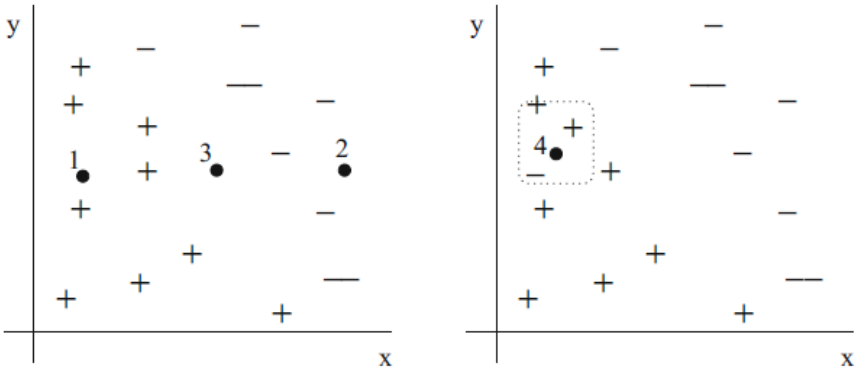
Table 2.2: Set of examples for nearest neighbor

Example	Shape	Crust Size	Crust Shade	Filling Size	Filling Shade	Class	#differences
X	Square	Thick	Gray	Thin	White	?	-
E1	Circle	Thick	Gray	Thick	Dark	Pos	3
E2	Circle	Thick	White	Thick	Dark	Pos	4
E3	Triangle	Thick	Dark	Thick	Gray	Pos	4
E4	Circle	Thin	White	Thin	Dark	Pos	4
E5	Square	Thick	Dark	Thin	White	Pos	1
E6	Circle	Thick	White	Thin	Dark	Pos	3
E7	Circle	Thick	Gray	Thick	White	Neg	2
E8	Square	Thick	White	Thick	Gray	Neg	3
E9	Triangle	Thin	Gray	Thin	Dark	Neg	3
E10	Circle	Thick	Dark	Thick	White	Neg	3
E11	Square	Thick	White	Thick	Dark	Neg	3
E12	Triangle	Thick	White	Thick	Gray	Neg	4

This algorithm may face some difficulties trying to label examples placed too close to borders of classes or affected by noise. In Figure 2.3 (extracted from Kubat, 2017) two plots are shown for a set of examples characterized in a bidimensional domain.

Examples are labeled as positive or negative. The algorithm is applied to objects numbered from 1 to 4. In the first plot, objects 1 and 2 are easily classified. They are surrounded by positive and negative examples respectively, so they are assigned to these classes. On the other hand, object number 3 is close to both positive and negative examples, which means that a small amount of noise may be decisive to decide whether it is positive or negative. Thus, the decision for this example is not reliable.

Figure 2.4: Plots of examples in a bidimensional domain



The second plot illustrates class noise. Object 4 is placed in a region where it is supposed to be surrounded by positive objects, but the nearest neighbor is negative. This means that with a classifier with k=1, the decision would be negative, and with k=2, there would be a tie. Choosing k=3 would make sure that the choice is positive, since the votes would be two positive against one negative.

In this work, results for k-nn are evaluated through the concepts of accuracy, precision (different to the one for decision trees), recall and specificity:

- *Accuracy*: it is, again, the ratio of good predictions.
- *Precision*: it is the ratio of positive instances correctly predicted over the total number of positives predicted.
- *Recall*: it measures the number of activations detected correctly over the total number of real positives.
- *Specificity*: it consists of the correctly number of negatives predicted over the total number of negative measures.

2.3.3 Cross-validation

Cross-validation is a statistical method that evaluates and compares learning algorithms by dividing data into two parts: training data to learn a model, and testing data to validate this model. Then, it changes (i.e. crosses) both in successive rounds, so each data point has a chance to be

validated against (Refaeilzadeh, 2009). The basic form is k-fold cross-validation, which is the one used in this work with k=10.

K-fold cross-validation uses partitions of k segments as equal as possible. Then, k iterations of training and validation are performed, using one partition for validation and the other k-1 for training. The main advantage of using k-fold cross-validation for this work is avoiding the model being adjusted to very specific characteristics of the training data.

2.3 Tools

In this section, we explain the two main tools used in this work. In first place, R, which is the language used to implement the algorithms considered in section 2.1. After that, RStudio, which is the Integrated Development Environment (IDE) used to work with R.

2.3.1 R

R is a scripting language for statistical data manipulation and analysis. Its working environment is open source, which means that all its libraries and resources have free access (Matlof, 2001).

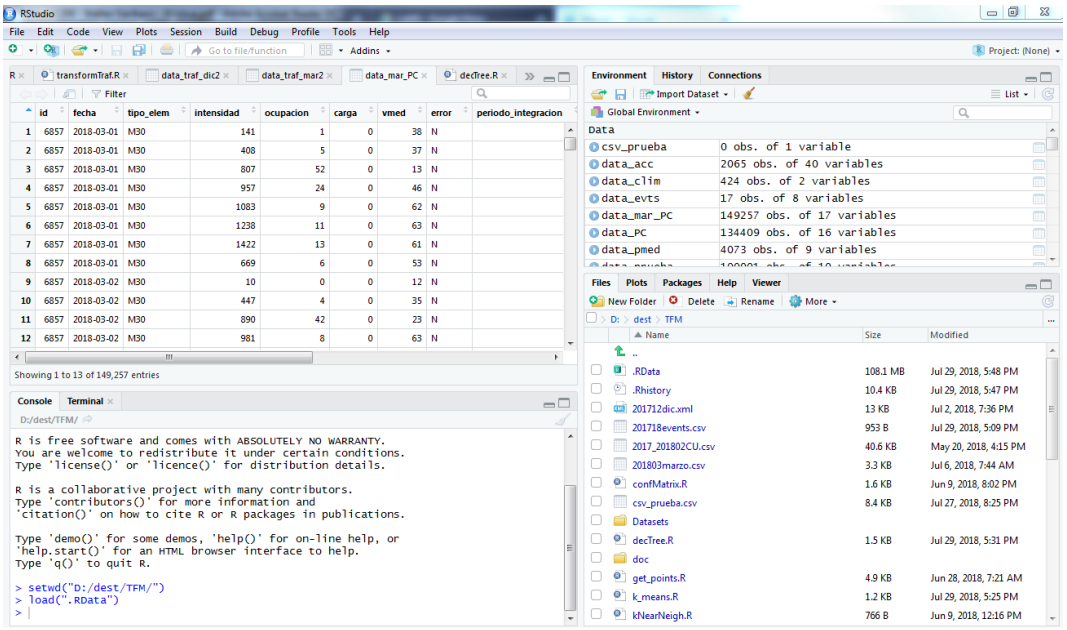
As a scripting language, R has some characteristics worth to mention. It incorporates characteristics of object-oriented and functional programming languages. It is polymorphic, which means that functions can be used with different type of input objects. Iterative operations can be done implicitly using R's functional features, which makes them much more efficient in terms of computational cost and time.

R's open-source nature favors the existence of a big community in Internet which makes easier for users to find useful add-ons and help for their problems in different forums. This is one of the main reasons that has make it so popular.

2.3.2 RStudio

The basic R environment works in a command window, so users need to type there their commends and examine the results with little convenience support. Many tools have been developed in order to do R programming more user-friendly. The Integrated Development Environments (IDEs) are Graphical User Interfaces (GUIs) oriented to programming. The one used in this work is RStudio. Figure 2.4 shows its main window.

Figure 2.5: RStudio interface



Initially, the main window is divided in four. The top left window is aimed at the visualization of all kind of files and objects. The bottom left window is the console, where R commands and functions can be executed. The top right one shows the environment (defined variables and functions), history (commands previously executed) and connections to remote servers. The bottom right one shows the folder that contains the workspace and all the files that may be used or visualized. Many more windows can be shown in the main one. The view is fully customizable. Besides, RStudio includes an editor supporting direct code execution and tools for plotting.

2.4 Conclusions from Chapter 2

In this Chapter, the state of the art, tools and techniques used in this work have been presented. We have discussed the different kinds of traffic congestion that exist and how identify them.

We have presented the basics of the Machine learning techniques used to extract value from the available information and to perform the short term predictions. Since we need to split the information between training and testing data in order to apply Supervised Learning algorithms, we have presented Cross-validation as a reliable method to do this. Finally, a brief introduction to the programming language R, and the IDE R is presented.

Chapter 3 - Data sources, acquisition and processing

This work uses several data sources to acquire the information required for the planned tasks. The main ones are public open data. Open data stand for sets of data that are not tied to copyright or patents and everyone has free access to them. Additional data on events have been specifically created for this work. In this chapter, the content, format and processing of the files containing the data is explained.

3.1 Traffic data

These data were acquired using the open data webpage maintained by Madrid's government: https://datos.madrid.es/portal/site/egob/. This webpage contains useful information regarding different aspects of the city. It supports the acquisition of data in real time, though historical data have been used in this work.

The information on traffic consists of series of measures for locations distributed around the city. The time interval of these measures is 15 minutes. The file format is Comma Separated Values (CSV). There is a different file for the data corresponding to each month-year. Table 3.1 shows a short set of measures. The columns are:

- The *id* is the unique identifier of the point of measure in the control systems of Madrid's government.
- *Fecha* is the time and date of the measure.
- *Tipo_elem* is the kind of point of measure. It indicates if the point belongs to M30 ring road or if it is an urban point.
- *Intensidad* is the intensity of the point in vehicles per hour in the previous 15 minutes.
- *Ocupacion* is the occupation time of the point in the previous 15 minutes in percentage.
- *Carga* is the road load. It is a parameter depending on intensity, occupation time and the capacity of the road where the point is located, i.e. the grade of use. It defines the level of use of the road in percentage.

- *Vmed* is the average speed of the vehicles in the road for the past 15 minutes in kilometers per hour, only for M30 points.
- *Error* indicates if there has been any error in the measure. Its values are N for no errors, E when the quality parameters of the measures are not optimum, and S if any of the measures taken is wrong and has not been integrated.
- *Periodo_integracion* is the number of measures taken for the integration period.

Table 3.1: Traffic data

id	Fecha	tipo_elem	intensidad	ocupacion	carga	vmed	error	periodo_integracion
1001	01/03/2018 0:00	M30	720	26	0	53	N	4
1002	01/03/2018 0:00	M30	540	2	0	68	N	4
1003	01/03/2018 0:00	M30	600	2	0	76	N	4
1006	01/03/2018 0:00	M30	690	2	0	59	N	4
1009	01/03/2018 0:00	M30	510	1	0	66	N	4
1010	01/03/2018 0:00	M30	1080	1	0	56	N	4
1011	01/03/2018 0:00	M30	585	2	0	55	N	4
1012	01/03/2018 0:00	M30	405	4	0	51	N	4
1013	01/03/2018 0:00	M30	705	2	0	69	N	4
1014	01/03/2018 0:00	M30	405	2	0	51	N	4
1015	01/03/2018 0:00	M30	675	1	0	69	N	4
1016	01/03/2018 0:00	M30	255	1	0	58	N	4
1017	01/03/2018 0:00	M30	1020	1	0	68	N	4
1018	01/03/2018 0:00	M30	1350	3	0	65	N	4

There were two main issues that made the data acquired from this source inconsistent. Firstly, there were some incorrect measures, pointed out by the *error* column. The measures that did not have as value associated for this column *N* were discarded. Secondly, many of the urban

points had as value for *vmed* 0. This happens because the data acquisition devices in urban points do not have a system for speed control. In this case, the data preprocessing assigned to these measures the maximum speed allowed (50 kilometers per hour) and, in the later processing (see Chapter 4), *vmed* was not used.

The amount of data for this aspect is really large. The CSV file containing the data of one month is around 600 MB and a measure for each point every 15 minutes is too detailed for this study. Hence, data must be filtered and processed to be adapted to our needs. Even though the initial prevision was to use data from, at least, one full year, finally, because of matters of memory and computational cost, only the months of December 2017 and March 2018 were used. These months were selected because they had plenty relevant events, so they are illustrative for this work. There is activity in stadiums with league matches played in weekends and European ones between Tuesdays and Thursdays. December includes the Christmas shopping period. That March also had many rainy days, which makes it good to study the influence of rainfalls in urban traffic congestions.

3.2 Measure points

The traffic data is organized by the number of identification of the measure points. These points have a geographical position. They are stored in an independent CSV file, which was acquired from the same source, the Madrid's government open data webpage. Table 3.2 shows some of the data contained in this CSV file. Its columns are:

- *Id* is the identifier of the measure point. This is the same number that can be seen in the traffic data file. *Nombre* is the name of the measure point.
- *Tipo_elem* indicates if the measure point belongs to M30 or it is placed in another road or street.
- *X* and *Y* give the coordinates in units of the Universal Transverse Mercator (UTM) system of transverse coordinates.

Table 3.2: Measure points data

id	nombre	tipo_elem	x	y
6779	PM31452	M-30	438847,856	4469073,6
6691	PM12121	M-30	437426,505	4476616,9
6938	PM31851	M-30	435084,322	4472331,08
6719	PM20233	M-30	443364,217	4479753,76
6737	PM20721	M-30	444088,825	4474829,88
3559	PM22392	M-30	436693,686	4479233,13
6720	PM20261	M-30	443554,743	4479247,59
6756	PM21081	M-30	442723,212	4471643,02
6949	PM31891	M-30	432749,973	4470038,75
3821	PM20025	M-30	442925,121	4481473,98
6769	PM22471	M-30	436432,204	4479965,66
6848	15RM08PM01	M-30	439107,195	4472320,71
6685	PM11201	M-30	441891,725	4470593,42
6861	15RR60PM02	M-30	439563,447	4472421,14

3.3 Weather data

The source used for these data is the official webpage of AEMET (the Spanish Weather Agency), which has a section dedicated to open data: http://www.aemet.es/es/datos_abiertos/AEMET_OpenData.

In this case, the information is provided using web services in JavaScript Object Notation (JSON) files. These are converted to CSV to work with them in R. Table 3.2 shows some example measures for the station placed in the "Ciudad Universitaria" area. This corresponds to the Moncloa campuses of several universities in the city of Madrid. The columns are:

- *Fecha* is the day when the measures were taken.
- *Alt* is the altitude of the station in meters above the level of the sea.
- *Prec* is the quantity of rain in liters per square meter.
- *Tmed* is the average temperature measured in the day, followed by the minimum, *Tmin*, the maximum, *Tmax*, and the times when each one has been measured, *Horatmin* and *Horatmax*.
- *Dir* is the direction of the strongest gust of wind of the day in tens of degree.
- *Velmedia* is the average speed of wind in meters per second.

- *Racha* is the maximum speed of a gust of wind in the day, and *horaracha* is the time of the day when it was measured.

Table 3.3: Weather data

Fecha	Alt	prec	Tmed	Tmin	Horatmin	Tmax	Horatmax	Dir	Velmedia	Racha	horaracha
04/06/2017	664	0	20,7	13,8	23:59	27,6	12:00	26	3,6	11,9	13:10
05/06/2017	664	0	19,3	10,2	6:00	28,4	16:00	25	1,7	10	15:50
06/06/2017	664	0	22,2	14,7	4:50	29,7	16:00	28	2,2	10,6	18:20
07/06/2017	664	0	22,8	13	3:50	32,7	14:40	17	1,1	6,7	22:30
08/06/2017	664	0	26,3	19	5:40	33,6	15:00	27	1,4	7,8	15:20
09/06/2017	664	0	26,4	18,9	6:00	34	14:20	25	1,1	9,4	14:50
10/06/2017	664	0	25,6	15,9	4:30	35,3	16:40	24	1,7	5,6	12:10
11/06/2017	664	0	29,1	21,2	5:20	37	14:20	23	1,7	6,9	13:40
12/06/2017	664	0	30	22,8	5:50	37,2	15:30	14	1,1	8,9	22:50
13/06/2017	664	0	29	21,3	5:00	36,7	16:00	11	1,4	7,5	2:20
14/06/2017	664	0	30,2	21,9	5:10	38,4	15:50	24	2,2	10	14:40
15/06/2017	664	0,5	30,9	21,2	4:50	40,6	15:40	12	1,7	15,6	19:50
16/06/2017	664	0	29,3	20	5:00	38,6	16:00	7	1,7	8,6	23:59
17/06/2017	664	0	33	24,8	6:00	41,1	15:00	6	2,2	9,2	0:10

3.4 Events

As section 1.2 stated, the main goal of this work is to extract added value analyzing and processing the traffic data from the city of Madrid. This is obtained considering events that affect directly traffic congestions somehow.

The considered events were some involving crowds, particularly the football matches of Real Madrid C. F. and C. Atlético de Madrid, and the shopping days around Christmas, and weather events such as storms. The weather data were obtained from AEMET, as explained in section 3.3, and the data concerning the days around Christmas can be filtered directly in the traffic data file form dates. Thus, only information on matches was required. A new file for these events was created as shown in Table 3.4.

The columns in the events file are the following:

- *Event* identifies each different event.

- *Description* indicates the club hosting the match (RM for Real Madrid, ATM for Atlético de Madrid).
- *Date*, *Starting time* and *Ending time* gives the slot of time in which the game was played.
- *X* and *Y* give the coordinates in units of the UTM system of the match place.
- *Franja* indicates the intervals of time that are affected. The traffic data are divided in time intervals as section 3.5 will explain.

Table 3.4: Events data

Event	Description	Date	Starting time	Ending time	x	y	franja
1	RM	06/12/2017	20:45:00	22:45:00	441634	4478271	7
1	RM	06/12/2017	20:45:00	22:45:00	441634	4478271	8
2	ATM	02/12/2017	16:15:00	18:15:00	449158	4476336	5
2	ATM	02/12/2017	16:15:00	18:15:00	449158	4476336	6
3	RM	09/12/2017	16:15:00	18:15:00	441634	4478271	5
3	RM	09/12/2017	16:15:00	18:15:00	441634	4478271	6
4	ATM	16/12/2017	20:45:00	22:45:00	449158	4476336	7
4	ATM	16/12/2017	20:45:00	22:45:00	449158	4476336	8
5	RM	23/12/2017	13:00:00	15:00:00	441634	4478271	5
6	RM	03/03/2018	20:45:00	22:45:00	441634	4478271	7
6	RM	03/03/2018	20:45:00	22:45:00	441634	4478271	8
7	ATM	11/03/2018	16:15:00	18:15:00	449158	4476336	5
7	ATM	11/03/2018	16:15:00	18:15:00	449158	4476336	6
8	RM	18/03/2018	20:45:00	22:45:00	441634	4478271	7
8	RM	18/03/2018	20:45:00	22:45:00	441634	4478271	8
9	ATM	08/03/2018	20:00:00	22:00:00	449158	4476336	7
9	ATM	08/03/2018	20:00:00	22:00:00	449158	4476336	8

3.5 Processing and merging

Several operations were developed in R to integrate the information from the different sources. This processing is summarized next.

Due to the big size of the files containing the information of the traffic, it was decided to use the *data.table* library for R developed in the CRAN project. This library extends the R *data.frame* and is focused on the aggregation of large data. Using this library, it is possible to

perform standard operations (add/modify/delete) by group without the need of copying, which reduces time and computational costs.

Our analysis needs to reduce the number of measures to consider, grouping them in time intervals and performing operations in their numerical values. Besides, regarding to prediction, it is required to stablish a criteria to consider when a traffic congestion appears. This activation is included in the *data.table* object. This process is explained in this section.

Figure 3.1: *data.table* containing the traffic data

	id	fecha	tipo_elem	intensidad	ocupacion	carga	vmed	error	periodo_integracion	hora	prec	color	franja	x	y	act
1	1001	2017-12-01	PUNTOS MEDIDA M-30	384	23	0	68	N	5	00:00:00	0	green	1	437146	4473498	0
2	1001	2017-12-01	PUNTOS MEDIDA M-30	408	20	0	67	N	5	00:15:00	0	green	1	437146	4473498	0
3	1001	2017-12-01	PUNTOS MEDIDA M-30	336	24	0	69	N	5	00:30:00	0	green	1	437146	4473498	0
4	1001	2017-12-01	PUNTOS MEDIDA M-30	324	24	0	64	N	5	00:45:00	0	green	1	437146	4473498	0
5	1001	2017-12-01	PUNTOS MEDIDA M-30	435	22	0	67	N	4	01:00:00	0	green	1	437146	4473498	0
6	1001	2017-12-01	PUNTOS MEDIDA M-30	300	25	0	75	N	4	01:15:00	0	green	1	437146	4473498	0
7	1001	2017-12-01	PUNTOS MEDIDA M-30	228	12	0	67	N	5	01:30:00	0	green	1	437146	4473498	0
8	1001	2017-12-01	PUNTOS MEDIDA M-30	336	21	0	61	N	5	01:45:00	0	green	1	437146	4473498	0
9	1001	2017-12-01	PUNTOS MEDIDA M-30	156	20	0	40	N	5	02:00:00	0	green	1	437146	4473498	1
10	1001	2017-12-01	PUNTOS MEDIDA M-30	96	10	0	42	N	5	02:15:00	0	green	1	437146	4473498	0
11	1001	2017-12-01	PUNTOS MEDIDA M-30	72	10	0	29	N	5	02:30:00	0	green	1	437146	4473498	1
12	1001	2017-12-01	PUNTOS MEDIDA M-30	168	23	0	58	N	5	02:45:00	0	green	1	437146	4473498	0
13	1001	2017-12-01	PUNTOS MEDIDA M-30	84	3	0	61	N	5	03:00:00	0	green	1	437146	4473498	0
14	1001	2017-12-01	PUNTOS MEDIDA M-30	96	2	0	50	N	5	03:15:00	0	green	1	437146	4473498	0
15	1001	2017-12-01	PUNTOS MEDIDA M-30	90	8	0	43	N	4	03:30:00	0	green	1	437146	4473498	0
16	1001	2017-12-01	PUNTOS MEDIDA M-30	180	19	0	71	N	4	03:45:00	0	green	1	437146	4473498	0

The function *parseTraf.R* reads and aggregates the data is done. Appendix A includes its complete code., Figure 3.1 shows a small set of rows of the resulting *data.table* object. This function works as follows:

- Reads the traffic data CSV file using *data.table* function *fread.*
- Splits date and time in two columns.
- Reads the weather data CSV file.
- Adds the daily quantity of rainfalls to the corresponding days.
- Defines time intervals in a new column.
- A new column is defined to indicate if there have been issues in the traffic. This is the *activation* introduced in section 2.2. An activation for an interval and point of measure is set if one of the following happens (the choice of these values is explained in section 4.2.4):

- *Ocupacion >= 60*
- *Carga >= 60*
- *Vmed <= 40*

- Geographical positions of the measure points are added using the corresponding CSV file.

The resulting *data.table* is still too large to be handled with our computational constraints. Moreover, it contains too detailed data for the intended analysis, so it can be simplified. This object needs to be transformed before algorithms can be applied on it. A new function called *transformTraf.R* is created. Its code can be seen in the Appendix A. This function gets a single measure for one interval and measure point by performing the following operations:

- It calculates the arithmetic mean for *intensidad, ocupacion, carga* and *vmed*.
- If there has been any activation (*act*=1) in any of the 15-minute measures in the interval, it will be considered as an activation for the whole interval.
- A new column *act_pred* is defined. **It represents the congestion status for the next traffic interval, this is, the value of "*act*" in the next interval**. This value is not the prediction, but the actual congestion value that will be the target of the prediction. It will be used as a label to identify traffic situations and recognize which values are related with future traffic congestions, i.e. as the label for the classification.

Figure 3.2: *data.table* containing transformed traffic data

	id	fecha	tipo_elem	intensidad	ocupacion	carga	vmed	error	periodo_integracion	hora	prec	color	franja	x	y	act_pred
1	1001	2018-03-01	M30	720	26	0	53	N	4	00:00:00	17.6	green	1	437146	4473498	0
2	1001	2018-03-01	M30	720	26	0	53	N	5	04:00:00	17.6	green	2	437146	4473498	0
3	1001	2018-03-01	M30	720	26	0	53	N	4	07:30:00	17.6	green	3	437146	4473498	0
4	1001	2018-03-01	M30	1840	17	0	57	N	4	09:00:00	17.6	red	4	437146	4473498	0
5	1001	2018-03-01	M30	2907	9	0	61	N	4	12:00:00	17.6	red	5	437146	4473498	0
6	1001	2018-03-01	M30	2642	8	0	63	N	5	16:00:00	17.6	red	6	437146	4473498	0
7	1001	2018-03-01	M30	2572	9	0	62	N	5	17:30:00	17.6	red	7	437146	4473498	0
8	1001	2018-03-01	M30	1497	13	0	62	N	5	20:00:00	17.6	yellow	8	437146	4473498	1
9	1001	2018-03-02	M30	240	1	0	52	N	5	00:00:00	16.9	green	1	437146	4473498	1
10	1001	2018-03-02	M30	1190	5	0	60	N	4	04:00:00	16.9	red	2	437146	4473498	0
11	1001	2018-03-02	M30	4116	20	0	56	N	5	07:30:00	16.9	red	3	437146	4473498	0
12	1001	2018-03-02	M30	2484	13	0	64	N	4	09:00:00	16.9	red	4	437146	4473498	0
13	1001	2018-03-02	M30	2893	9	0	64	N	5	12:00:00	16.9	red	5	437146	4473498	0
14	1001	2018-03-02	M30	2908	8	0	61	N	5	16:00:00	16.9	red	6	437146	4473498	0
15	1001	2018-03-02	M30	2630	8	0	60	N	5	17:30:00	16.9	yellow	7	437146	4473498	0
16	1001	2018-03-02	M30	2428	7	0	60	N	5	20:00:00	16.9	red	8	437146	4473498	0

Finally, the information on events must be added to the traffic data. This is done with two new columns, *RM* and *ATM,* which have binary values. They correspond to Real Madrid (*RM*) or Atlético de Madrid (*ATM*). The corresponding variable is 1 for a measure point and a time interval if this time interval contains any window of time between one hour before the event and one hour later. For example, if there is a match at SB at 20:00 of 16th of December, RM of all points for that day for intervals 7 and 8 would be 1. To achieve this, another function is defined, called *parseEvts.R,* which is fully explained in the Appendix A. A set of rows of the resulting *data.table* object is shown in Figure 3.3. Its main actions are the following.

- It reads the events CSV file using *fread.*
- It assigns to each measure point and time interval the value of the columns *RM* and *ATM* as explained before.

Figure 3.3: *data.table* containing the transformed traffic data with the events added

	id	fecha	tipo_elem	intensidad	ocupacion	carga	vmed	error	periodo_integracion	hora	prec	color	franja	x	y	act	RM	ATM	act_pred
10334	1049	2018-03-02	M30	1558	6	0	58	N	5	16:00:00	16.9	yellow	6	443456.2	4472792	0	0	0	0
10335	1049	2018-03-02	M30	1402	6	0	59	N	5	17:30:00	16.9	yellow	7	443456.2	4472792	0	0	0	0
10336	1049	2018-03-02	M30	1002	4	0	63	N	5	20:00:00	16.9	yellow	8	443456.2	4472792	0	0	0	1
10337	1049	2018-03-03	M30	272	1	0	62	N	5	00:00:00	12.9	green	1	443456.2	4472792	1	0	0	1
10338	1049	2018-03-03	M30	191	1	0	55	N	5	04:00:00	12.9	green	2	443456.2	4472792	1	0	0	0
10339	1049	2018-03-03	M30	446	2	0	66	N	5	07:30:00	12.9	green	3	443456.2	4472792	0	0	0	0
10340	1049	2018-03-03	M30	804	3	0	64	N	5	09:00:00	12.9	green	4	443456.2	4472792	0	0	0	0
10341	1049	2018-03-03	M30	1190	5	0	61	N	4	12:00:00	12.9	yellow	5	443456.2	4472792	0	0	0	0
10342	1049	2018-03-03	M30	729	3	0	63	N	5	16:00:00	12.9	green	6	443456.2	4472792	0	0	0	0
10343	1049	2018-03-03	M30	1055	4	0	62	N	5	17:30:00	12.9	yellow	7	443456.2	4472792	0	1	0	0
10344	1049	2018-03-03	M30	960	4	0	62	N	5	20:00:00	12.9	green	8	443456.2	4472792	0	1	0	1
10345	1049	2018-03-04	M30	371	2	0	57	N	5	00:00:00	5.5	green	1	443456.2	4472792	1	0	0	1
10346	1049	2018-03-04	M30	152	1	0	56	N	5	04:00:00	5.5	green	2	443456.2	4472792	1	0	0	0
10347	1049	2018-03-04	M30	290	1	0	60	N	5	07:30:00	5.5	green	3	443456.2	4472792	0	0	0	0
10348	1049	2018-03-04	M30	506	2	0	65	N	5	09:00:00	5.5	green	4	443456.2	4472792	0	0	0	0
10349	1049	2018-03-04	M30	962	4	0	62	N	4	12:00:00	5.5	green	5	443456.2	4472792	0	0	0	0

Different files containing data were obtained from the AEMET webpage, as explained in section 3.3. From them, the daily quantity of rain is extracted and assigned to each measure. This assumes that rains are uniform in all points of the city through all the day, since only daily data were available.

Once all data has the proper format, the techniques explained in section 2.1 can be applied. This comes in Chapter 4.

3.6 Conclusions from Chapter 3

In this Chapter , the data gathered for this work have been discussed. We have presented the sources of information (Madrid's government Open Data webpage and AEMET Open Data webpage) and we have shown the operations needed in order to make these data functional for this work.

We have explained how to combine the sets of data to apply the algorithms explained in Chapter 2 and the need to simplify them because of the huge size and the limited computer resources. We have established time intervals in which we perform arithmetical means and other operations to make the data manageable by the algorithms developed.

Chapter 4 - Application of clustering and classification algorithms

This chapter explains the application of the algorithms in Chapter 2 to the problem of traffic congestions. First of all, it uses the k-means algorithm to divide the data in clusters with different characteristics. Given the size of the data acquired, this is done starting from three different points of view. First, k-means is applied to the whole set of data for each month to have an idea about what formats could have the different clusters and try to extract existing patterns. Second, the analysis adopts the hypothesis that the points more strongly affected would be geographically close to the points of the events. Thus, the algorithm is applied again establishing a maximum distance around the points of the events and filtering the measures. Third, as this assumption is not fulfilled (as section 4.1.2 will explain), the analysis applies the algorithm to all the points and time intervals in which the events are set to have influence since. This supports the preliminary analysis of data. After that, the chapter presents the development of models to predict traffic congestions for points of measures in the following time interval. This task checks alternatives with decision trees and k-nn. The information and conclusions of these analyses will be used to present the results in Chapter 5.

4.1 Application to the processed data by month

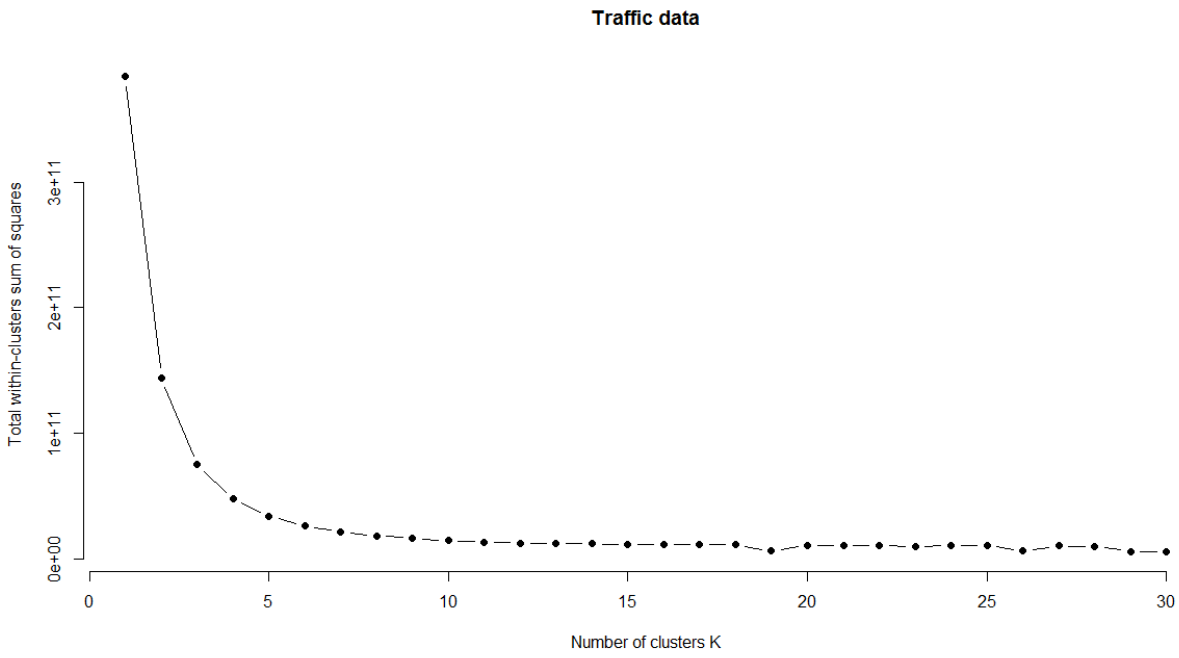
K-means is applied to the traffic data. Appendix B shows the functions applied. In order to obtain an estimation of the groups of data with differentiated properties, first, the work performs an analysis without geographical conditions. There is two datasets, for December 2017 and March 2018. This analysis uses the variables intensity, road load, occupation, quantity of rainfalls, and time interval to divide the set of data in clusters and try to recognize patterns in the measures.

The first step is to set the number of clusters that would be the optimum for the data to be divided. A division of a set of data in clusters needs to have a distance between them as big as possible and a distance between the elements within the clusters as small as possible. Using the output parameters of the R function *kmeans*, the Elbow method can be used to set a number of clusters, either optimum or good enough (Kodinariya & Makwana, 2013). This method consist in, given a bidimensional representation that has in the X axis the number of clusters and in the Y axis

the total distance of the elements within clusters between each other, choosing a value that gets a total distance small enough with a not too big number of clusters. Figure 4.1 shows this representation.

The *K-means* R function is used with the default algorithm (Hartigan and Wong, 1979) and the maximum number of iterations is set to 200. The variables that are used for the clustering in each measure are *intensidad, ocupacion, carga* and *franja*. Since *franja* has categorical values, the data are not previously escalated. *Vmed* is left out because there were some incorrect measures for average speed that needed to be corrected.

Figure 4.1: Elbow method for traffic data in December 2017



In Figure 4.1 we can see that setting the number of clusters to 5 is enough to get a distance small enough in comparison to dividing the data in more clusters. If we choose a higher number of clusters, the total distance between the elements within a cluster does not decrease substantially. It is maintained in an almost constant value. This trend is observed for values above 5. Thus, 5 is the chosen value. Figure 4.2 shows the sums of squares of the distances between the elements of

each cluster (*km$withinss*) and the sum of squares of the distances between clusters (*km$betweenss*).

Figure 4.2: Distance within and between clusters for traffic data in December 2017

```
> km$withinss
[1]  5057154696  5105153050 13252480575  5399422700  4952922332
> km$betweenss
[1] 350062339758
```

These are values in the order of ten to the power of nine for the distances between elements, which are really big. This is reasonable since the number of elements per cluster is, indeed, really big. Although the elements are close, the sum of squares of all distances between elements has to be big. Nonetheless, as shown in Figure 4.3, the ratio of the distances between clusters and the distances between points within a cluster is one order of difference. This means that there is some differentiation between clusters.

It could be of some use to take a look at the centers of the clusters in Figure 4.4. This figure shows the average values for each five variables of the measures assigned to each cluster. These five vectors act as geographical centers for each cluster in the 5-dimensional space where the clusters are placed.

Figure 4.3: Distance between clusters and distances within clusters ratio in December 2017

```
> km$betweenss/km$tot.withinss
[1] 10.38336
```

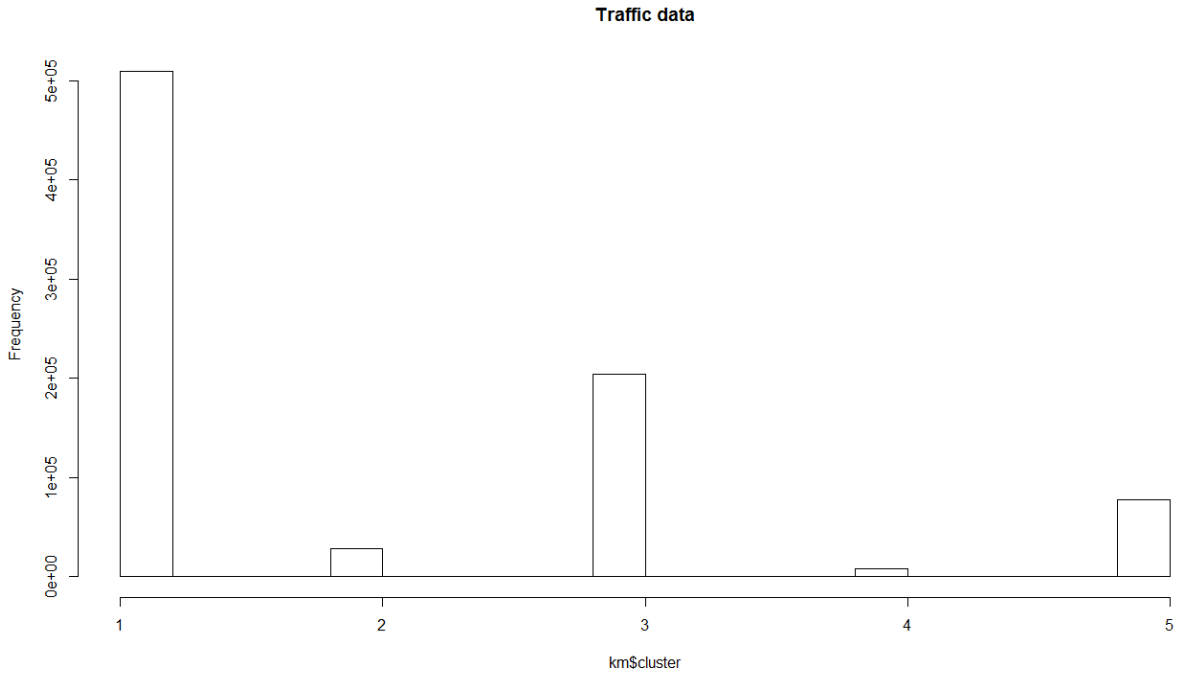
Figure 4.4: Cluster centers for traffic data in December 2017

```
> km$centers
  intensidad ocupacion    carga      prec   franja
1   143.9301  5.805391 14.05007 0.8618381 4.079682
2  2455.9673 10.994153 32.96387 0.3900886 5.480202
3   578.2229  8.014380 28.25008 0.5535120 5.264491
4  4429.6811 12.801311 33.79499 0.2798502 5.315469
5  1245.3982 10.756059 35.39734 0.3997257 5.321037
```

December 2017 did not have a remarkable quantity of rainfalls, so the values of *prec* of the centroids are small and their influence in the cluster division is limited. The time interval value for

all clusters but the number one is really similar in all four. Since it is the central value, it can be assumed that for bigger values of intensity, the time interval is the result of means of mostly 3 and 7 (which are the time intervals in which people go to or come from work) and for lower ones the values involved may be 1/2 and 5/8. The rest of the centers of the clusters has increasing values of intensity, road load and occupation except in the case of number 5, which has the biggest value for *carga*. This fact suggests that there might be a geographic component in this division. In roads like M30 or highways that come into the city like A-2, there can be a great number of vehicles per hour without the need of reaching big numbers in *carga* or *ocupacion*. On the other hand, smaller streets may hit lower values for intensity while road load and occupation may be high since average speed is lower and traffic lights may stop vehicles without congestion. Given this situation, there may be a majority of M30 points in clusters 2 and 4, and the rest will have a more balanced distribution. Figure 4.5 shows the number of elements assigned to each cluster.

Figure 4.5: Elements assigned to each cluster for traffic data in December 2017

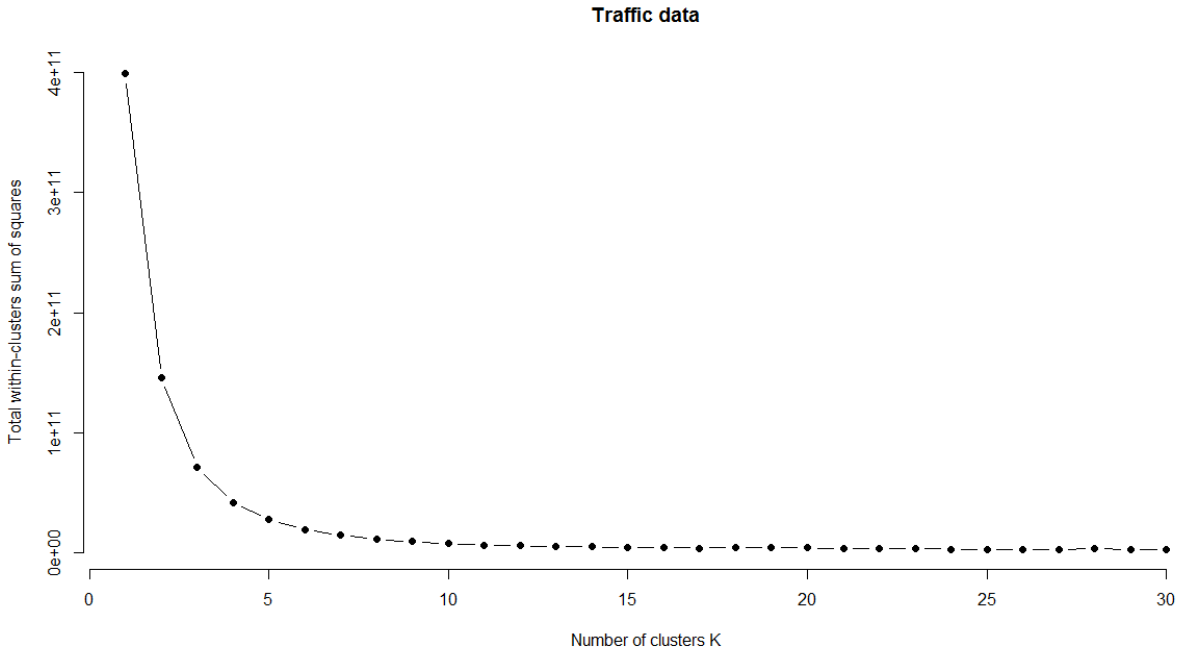


Distribution is not uniform at all. The cluster 1, which has the biggest number of measures assigned, is the one with the lowest values of congestion, since there are many more moments during the day when the traffic is fluid. This may imply that the distribution in this cluster is

balanced between M30 and urban measure points. The smallest column, corresponding to cluster number 4, is the one containing the most critical values of congestion. The clusters 2 and 5 are the ones more similar to each other. Nonetheless, there is a considerable difference in intensity between them which may lead to think that there are more M30 points of measure in 2 and both kinds are represented in 5 in more comparable proportions.

For the analysis of March, the Elbow method is applied and 5 clusters are selected. In Figure 4.6, it can be seen that the best suited value according to this method is 5.

Figure 4.6: Elbow method for traffic data in March 2018

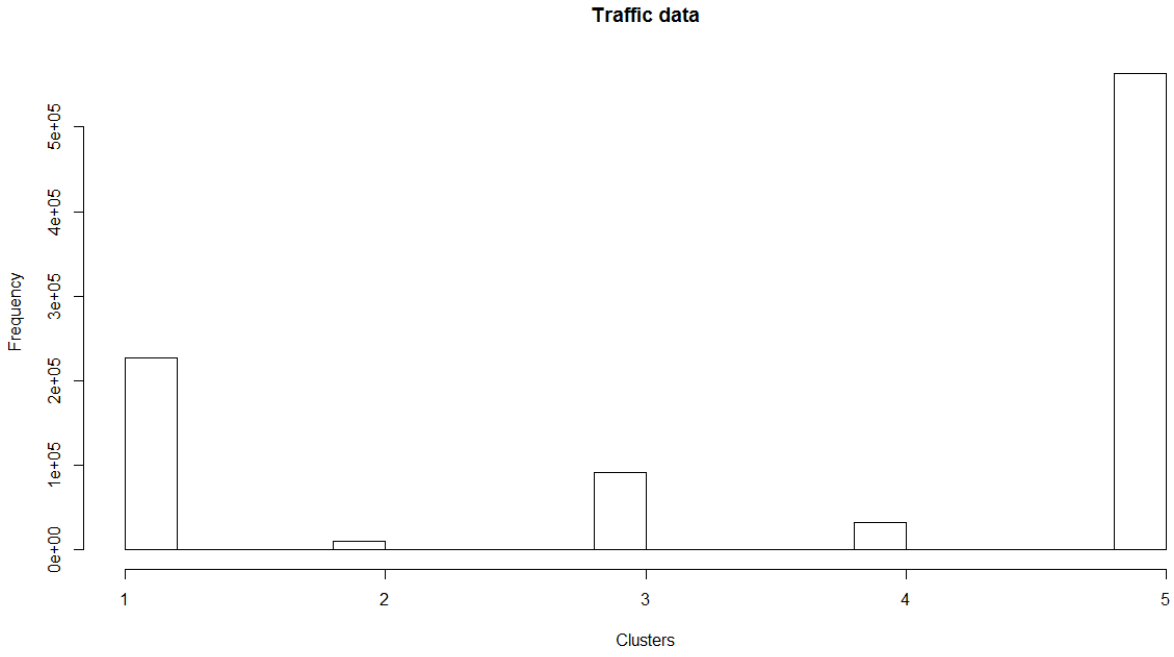


Figures 4.7 and 4.8 respectively show the centers for the clusters and the histogram for the number of measures by cluster. The pattern in the data of December is repeated. The biggest cluster is the one containing the lower values for intensity and road load, while the more critical values for traffic congestion correspond to the smaller clusters. Since March was a rainy month, it is worth to mention that the center values for rainfalls (*prec*) have greater values.

Figure 4.7: Cluster centers for traffic data in March 2018

```
> kmMar$centers
  intensidad ocupacion    carga     prec   franja
1   556.6150  8.111226 28.77654 4.820177 5.308858
2  4249.4106 13.240240 53.97089 4.770656 5.203068
3  1186.4305 11.233159 39.19362 4.895669 5.301786
4  2348.3962 11.831002 44.44221 4.922161 5.377765
5   137.5334  5.886666 13.82914 4.446905 4.088403
```

Figure 4.8: Number of measures by cluster for traffic data in March 2018



4.2 Application to the processed data by geographical area

Initially, it was decided to divide the data geographically in three areas in the city of Madrid as stated in section 4.1. Even though the application of clustering to the whole set of one month of data gives an idea of the different kinds of measures, it could add extra information to perform this analysis with data divided geographically. The hypothesis here is that points close to the places where the events are happening should be more prone to have traffic congestion problems. The initial analysis was done for December 2017 and then for March 2018.

4.2.1 Application to points close to the Santiago Bernabéu stadium

Euclidean distance is used to define the threshold to select the points arbitrarily close to the Santiago Bernabéu (SB) stadium considered in this analysis. The threshold chosen for the distance calculated through UTM coordinates is 2,000. This threshold is considerably lower than the one chosen for Wanda Metropolitano (see later in this section), since Bernabéu is placed deep in the city where the density in measure points is higher.

The dataset containing the information of points close to Santiago Bernabéu for December 2017 has 111,733 rows and contains the measures for 529 points. The functions used to build this dataset are shown in the Appendix B. Data are, again, not previously escalated since they use categorical data in *franja* and *RM*.

The Elbow method is applied again to choose the number of clusters. Figure 4.9 plots the distance between elements in a cluster and the number of clusters. Distances are still big as can be seen in the Y axis and in distances shown Figures 4.10 and 4.11, but 5 clusters are enough to assure that they are differentiated.

Figure 4.9: Elbow method for SB

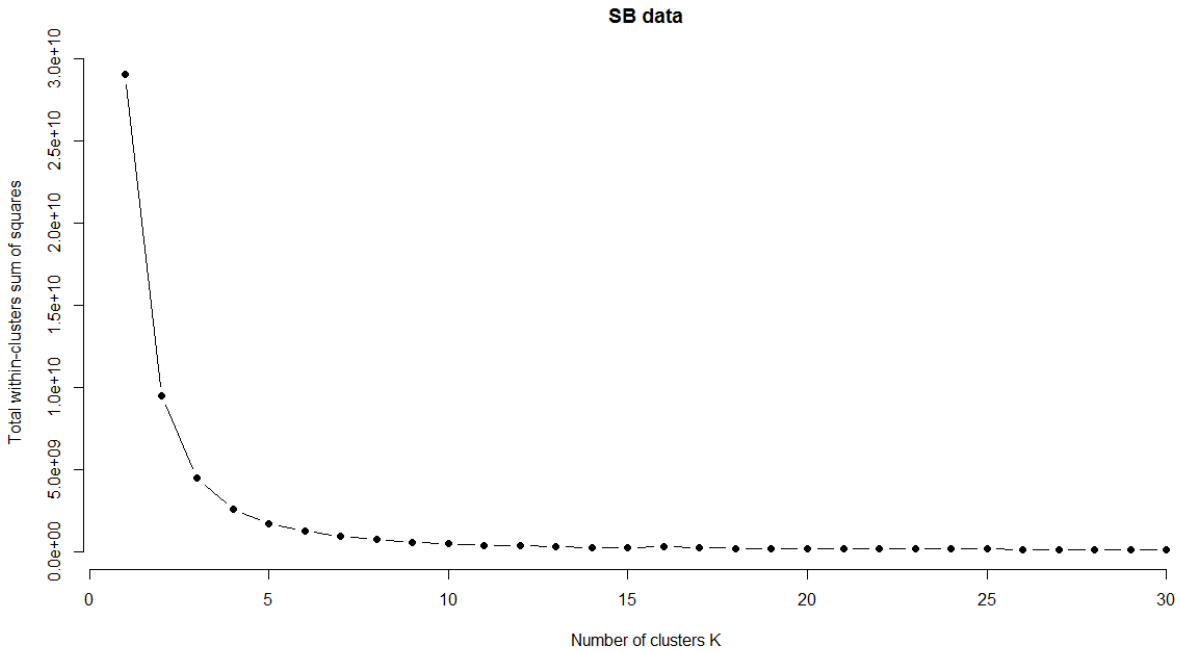


Figure 4.10: Distances within and between clusters for SB

```
> kmSBdic$withinss
[1] 360645740 447122319 314237270 302071995 325312192
> kmSBdic$betweenss
[1] 27272592011
```

Figure 4.11: Distances between and within clusters ratio for SB

```
> kmSBdic$betweenss/kmSBdic$tot.withinss
[1] 15.58978
```

Figure 4.12: Cluster centers for SB

```
> kmSBdic$centers
  intensidad ocupacion    carga      prec   franja          RM
1   128.7413  3.732574 10.45545 0.9076895 3.839791 0.011394447
2  2364.0555 13.985785 51.06446 0.1247603 5.223802 0.007603306
3   882.9705  9.029441 34.41888 0.3928228 5.379644 0.017374273
4  1443.8193 11.396353 42.39430 0.3156019 5.507678 0.014943789
5   449.7772  7.845196 24.64891 0.6897847 5.205185 0.017171939
```

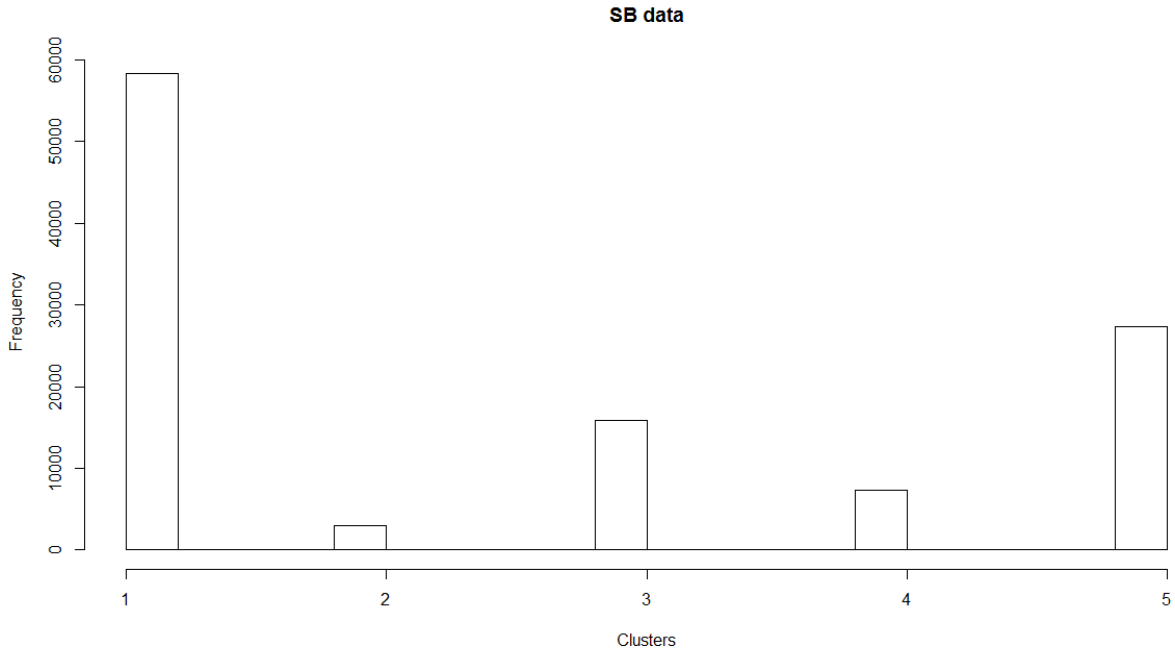
In Figure 4.12, the values for the centroids are shown. It can be seen that the measures with more critical values for traffic congestion are grouped in cluster 2. However, it is the cluster with the lowest values for precipitations and the less affected by intervals with football games. Figure 4.13 shows the ratio of intervals with RM=1 for cluster 2 and the total with RM=1.

Figure 4.13: Ratio of intervals affected by football matches in SB for cluster 2

```
> sum(data_SB_dic[kmSBdic$cluster==2,]$RM==1)
[1] 23
> sum(data_SB_dic$RM==1)
[1] 1540
> sum(data_SB_dic[kmSBdic$cluster==2,]$RM==1)/sum(data_SB_dic$RM==1)*100
[1] 1.493506
```

To calculate this ratio, we divide the number of measures belonging to cluster 2 by the total of measures affected by a football match at SB. Only 1.49 % of the measures affected by football matches are included in this cluster. This is a number really low considering the initial assumption that the closest points to the stadium would present measures with critical values for traffic congestion. Figure 4.14 shows, among other information, that cluster 2 has the lowest number measures.

Figure 4.14: Number of measures assigned to each cluster for SB



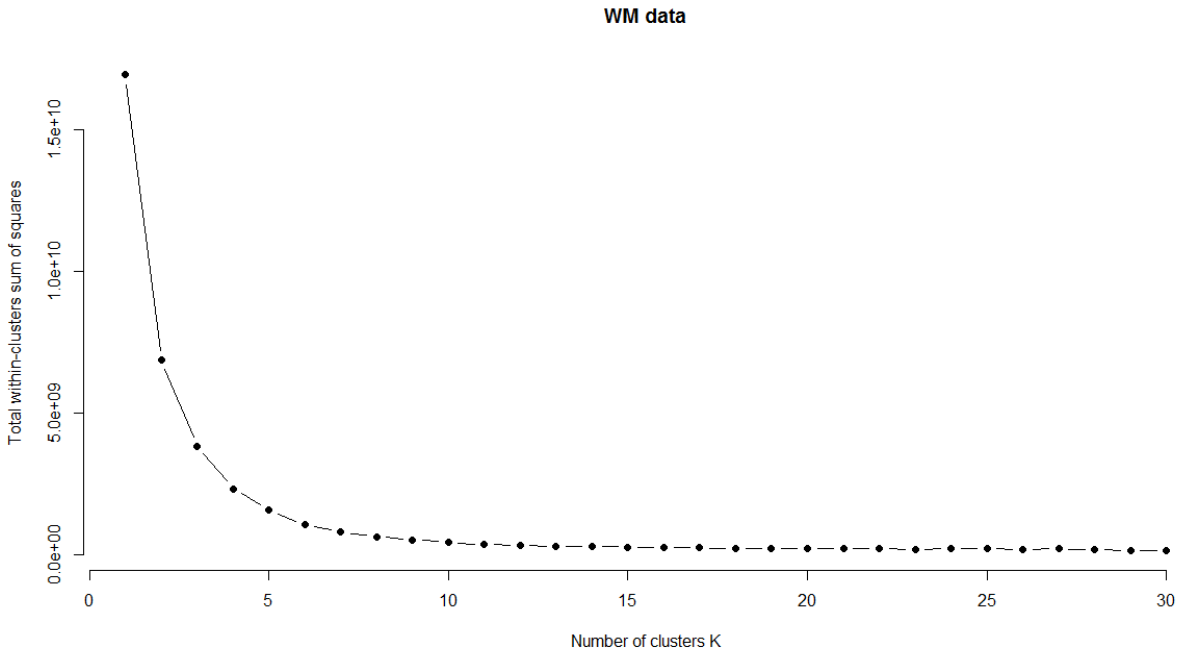
SB data follow a similar distribution to the whole set of data. The cluster with more critical values for traffic congestion has the lowest number of measures assigned (number 2), and the one with the less critical values (number 1) is the biggest one. The percentage of measures with *RM*=1 for the rest of clusters are: 43.12% for number 1, 30.45% for number 5, 17.88% for number 3 and 7.08% for number 4.

4.2.2 Application to points close to the Wanda Metropolitano stadium

Euclidean distance is used to calculate the threshold to select the points arbitrarily close to the Wanda Metropolitano (WM) stadium. The threshold chosen for the distance calculated through UTM coordinates is 5,000. WM is placed in the Rosas district, which is found in the outskirts of Madrid. There is an important highway and its ramifications passing nearby, the A-2 leading to Barcelona, and different industrial parks and bedroom towns. In this case, the threshold is bigger than for SB because the density of measure points is lower, and there is need to consider a bigger area to get representative data.

The dataset containing the information of points close to WM for December 2017 has 110,909 rows and contains the measures for 551 different points. The functions used to build this data set are fully explained in the Appendix B. Data are again not previously escalated, as they use categorical data in *franja* and *ATM*. Figure 4.15 depicts the plot of the application of the Elbow method.

Figure 4.15: Elbow method for points close to WM



Once again, with 5 clusters the results for kmeans application is good enough. In Figures 4.16 and 4.17 it can be seen that the distances fit the same assumptions as for SB.

Figure 4.16: Distances within and between clusters for WM

```
> kmwMdic$withinss
[1] 351456278 191582492 231200586 400104763 382970327
> kmwMdic$betweenss
[1] 15355980401
```

Figure 4.17: Distances between and within clusters ratio for WM

```
> kmwMdic$betweenss/kmwMdic$tot.withinss
[1] 9.860552
```

Figure 4.18: Cluster centroids for WM

```
> kmwMdic$centers
  intensidad ocupacion    carga      prec   franja        ATM
1   437.9787  7.187715 28.87184 0.5894421 5.464855 0.02556002
2  3644.6930 14.946203 46.32911 0.3803797 4.882911 0.01265823
3  1696.1716 13.792793 43.70175 0.1734471 5.293504 0.01375059
4   112.7226  6.155602 15.03899 0.8796285 3.928430 0.01469443
5   897.1880 10.803649 40.06490 0.2571748 5.399264 0.02437266
```

Figure 4.18 shows the values for the cluster centers. Like in the case of SB, the cluster with more critical values of traffic congestion is number 2, and it is the one where football games in the stadium has less influence. However, values of centers for intensity (*intensidad*) are bigger than in SB. This may happen because of the proximity of A2 and branches, where average speed is higher and more vehicles can go through in the same time. In Figure 4.19, the ratio of intervals with *ATM*=1 for cluster 2 and the total of *ATM*=1 is shown.

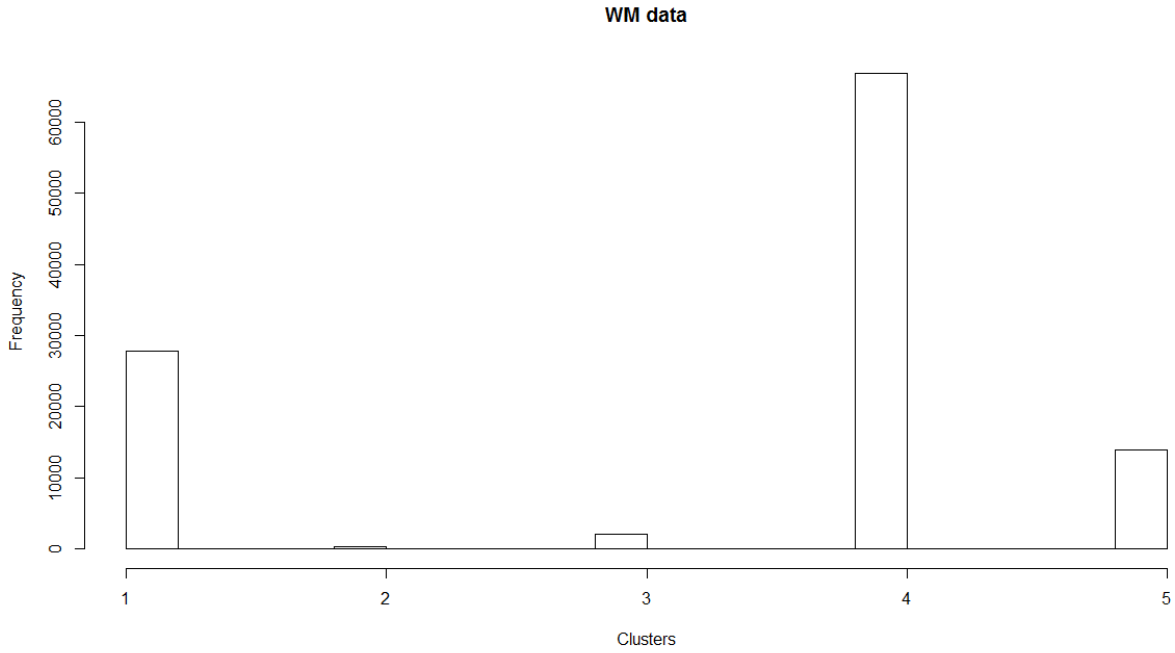
Figure 4.19: Intervals affected by football matches in WM

```
> sum(data_WM_dic[kmwMdic$cluster==2,]$ATM==1)
[1] 4
> sum(data_WM_dic$ATM==1)
[1] 2064
> sum(data_WM_dic[kmwMdic$cluster==2,]$ATM==1)/sum(data_WM_dic$ATM==1)*100
[1] 0.1937984
```

A total of only 0.19% of measures affected by football matches in WM are found in cluster 2. The results for the rest of the clusters are 34.50% for cluster 1, 1.41% for cluster 3, 47.53% for cluster 4, and 16.36% for cluster 5. Figure 4.20 shows the histogram of the distribution of the measures in each cluster.

Comparing the histograms for SB (see Figure 4.14) and WM (see Figure 4.20), though the ordering is different, it can be seen that the distribution is similar. Thus, conclusions can be similar.

Figure 4.20: Number of measures assigned to each cluster for WM

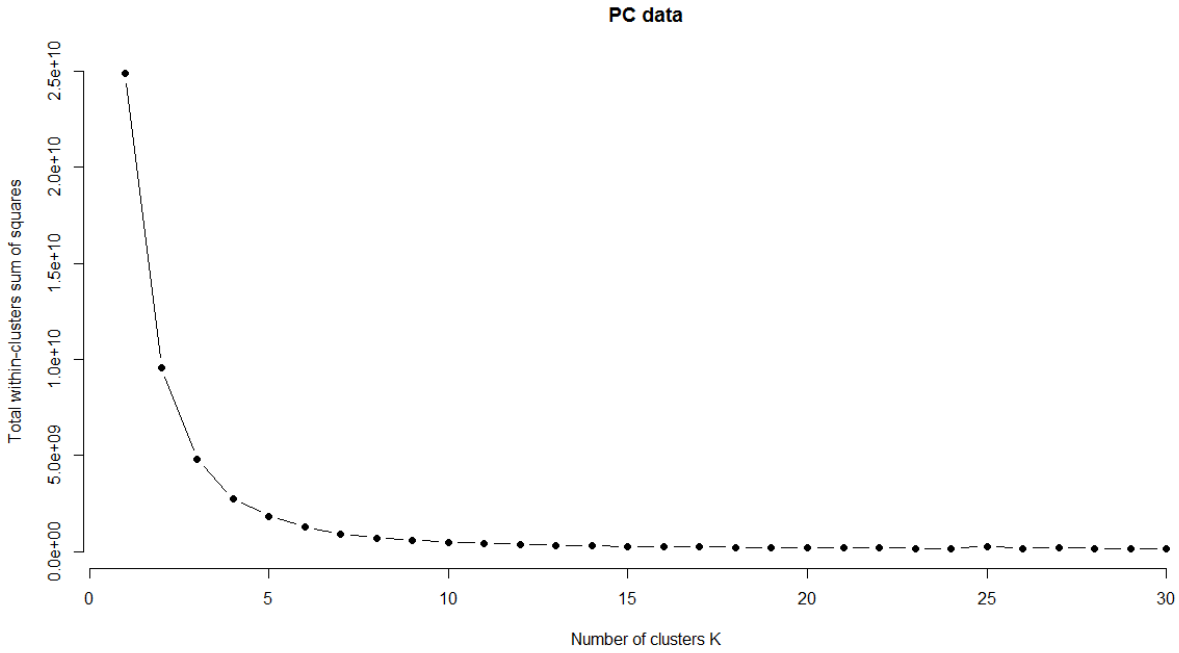


4.2.3 Application to points close to Plaza Callao

Euclidean distance is used to calculate the threshold for points arbitrarily close to Plaza Callao (PC). This exact point has been chosen because it is representative of Madrid's downtown. I gathers around many stores and points of interest that receive many visitors in the days around Christmas. The threshold chosen for the distance calculated through UTM coordinates is 2,500. In a similar way like SB, PC needs a lower threshold since it is found deep in the city, actually close to the geographical center of Madrid, and the density of points of measure is bigger.

The data set containing the information of points close to PC for December 2017 has 134,409 rows and contains the measures for 633 different points. Nonetheless, only data after 20th of December are taken as they are the most significant, what makes a total of 57,980 rows. The functions used to build this dataset are shown in the Appendix B. Data are, again, not previously escalated since they use categorical data in *franja*. Figure 4.21 depicts the plot for the Elbow method.

Figure 4.21: Elbow method for PC



The value for the number of clusters is 5 again, and the distances belong to the same order than in the previous applications of k-means. Figures 4.22 to 4.24 show the related data.

Figure 4.22: Distances within and between clusters for WM

```
> kmPCdic$withinss
[1] 330331518 482126362 332470408 339206404 369786556
> kmPCdic$betweenss
[1] 22989967815
```

Figure 4.23: Distances between and within clusters ratio for WM

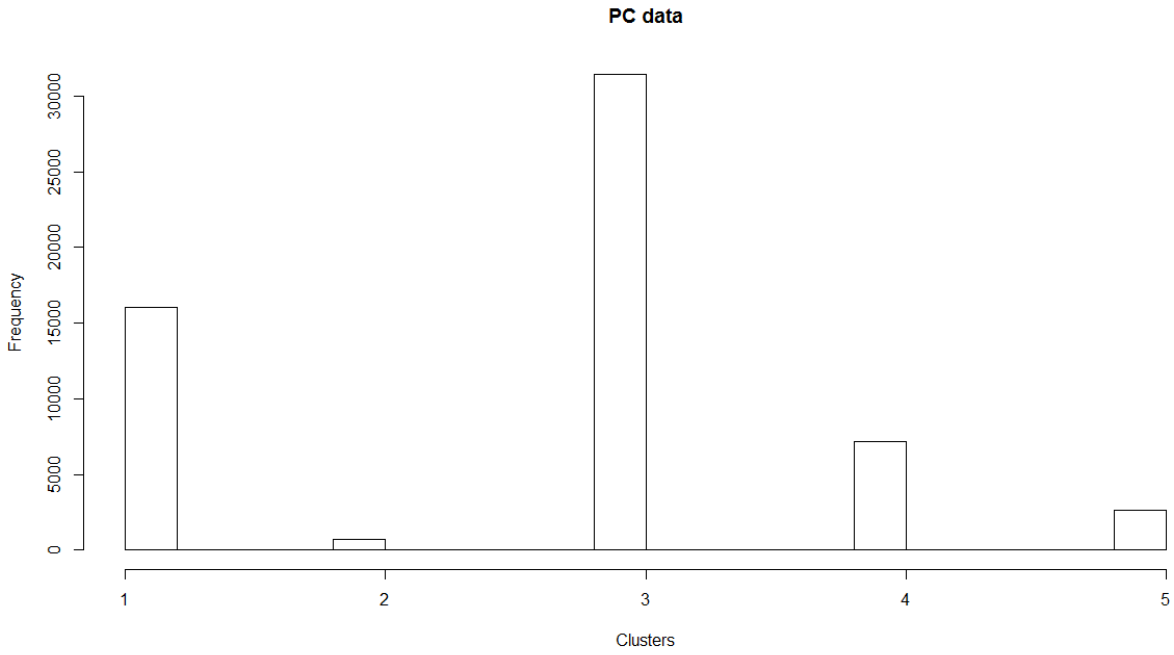
```
> kmPCdic$betweenss/kmPCdic$tot.withinss
[1] 12.40073
```

Figure 4.24: Cluster centroids for WM

```
> kmPCdic$centers
  intensidad ocupacion    carga      prec   franja
1   584.9583  7.779448 27.89206 0.4241764 5.051597
2  4021.9057 11.331536 10.51348 0.3132075 5.497305
3   165.2593  5.561783 15.36178 0.5381911 3.974172
4  1161.8259 10.243590 36.60061 0.3647575 5.425725
5  2143.7084 10.445330 32.80714 0.3743736 5.533789
```

As opposed to the two previous analysis, it can be seen that the center values of the intensity are higher for clusters 1 and 3, although surprisingly the value of the load of the road is lower. The number of measures by cluster is shown in Figure 4.25. In this case, the distribution is similar in terms of the percentage of measures for each cluster, extrapolating the centers of intensity.

Figure 4.25: Number of measures by cluster for PC



4.3 Application to the processed data by cases of study

The conclusions from section 4.2 are that the initial assumption that points close to the event zone are the ones that are getting bigger traffic congestion issues is not completely correct. Thus, next, an alternative analysis similar to the one in these sections is addressed. The main difference is that, instead of dividing the data geographically, only time intervals affected by the events for all points of measures are taken. This time, the data for both months in discussion are considered together, since the amount of data is already cut down by the time interval.

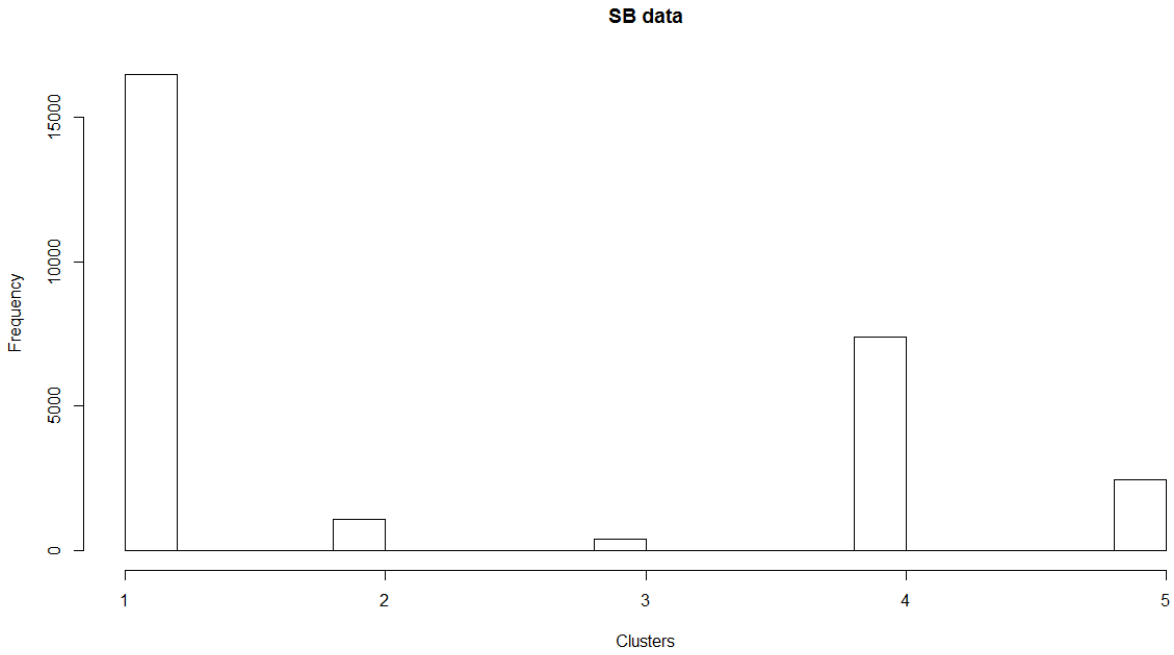
4.3.1 Application to time intervals affected by matches in the Santiago Bernabéu stadium

The time intervals that are affected by a match in SB give 27,767 measures for a total of 3,996 points. The application of the Elbow method implies again that 5 is a valid value to get differentiated clusters. The distances between clusters and between the elements within a cluster are separated by one order of magnitude, like in the previous cases, what means that the choice of 5 with the Elbow method is correct. Figure 4.26 shows the centers of the clusters.

Figure 4.26: Centers for intervals affected by matches in SB

```
> kmSB2$centers
  intensidad ocupacion    carga     prec   franja
1   183.4401  6.806907 17.55071 3.501070 6.615637
2  2283.1548  7.474513 25.90639 3.443188 6.667285
3  3778.9770  8.678571 35.02041 3.389541 6.492347
4   601.9827  6.419997 25.83498 3.848503 6.553042
5  1271.8780  7.356709 27.24443 3.540049 6.608431
```

Figure 4.27: Number of measures per cluster for time intervals affected by matches in SB

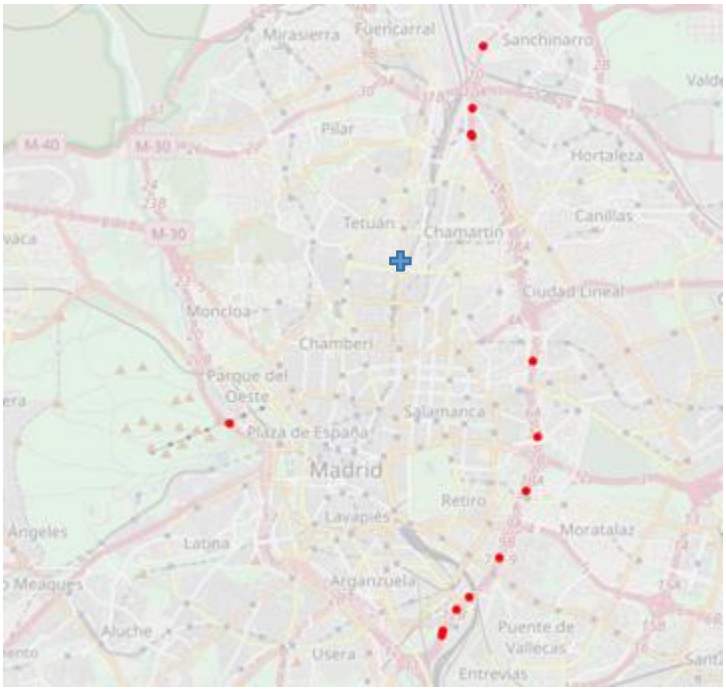


There is explicitly one cluster with critical values of congestion. Given that March was a rainy month, the centers for rainfalls take values above 0. Since most matches happen in the

afternoons or evenings, the centers for the intervals are found between 6 and 7 hours. Figure 4.27 shows the histogram of the measures in each cluster.

The distribution is similar to previous applications. However, this time the number of measures is much lower, so it is worth to take a look to cluster number 3, which includes the critical ones. If, for example, an arbitrary limit of 80% of road load is set to have critical traffic congestions, the measure points affected are shown in the map of Figure 4.28 as red points.

Figure 4.28: Points with critical values of road load with matches at SB in cluster 3



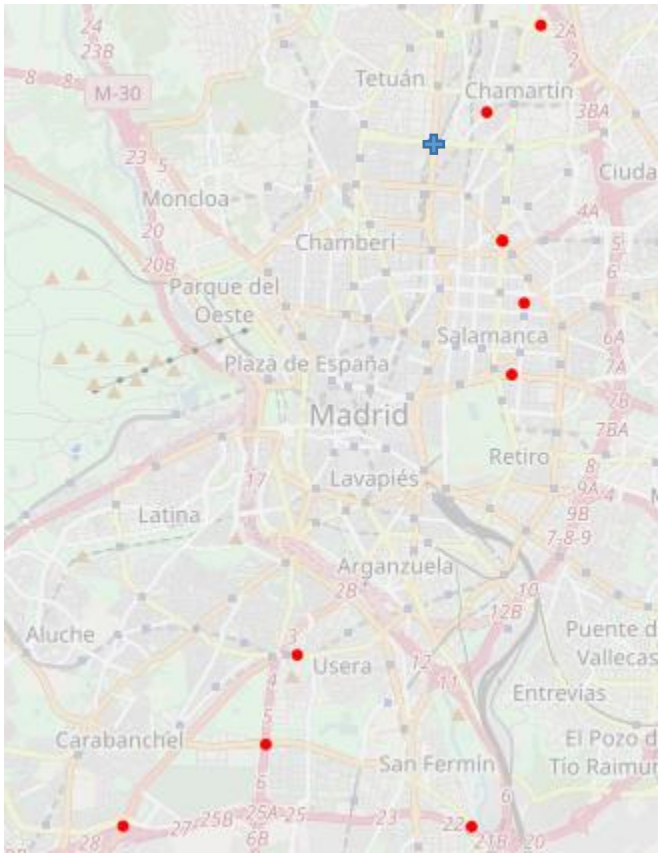
SB is placed in the Chamartín district, where Paseo de la Castellana and Avenida de Concha Espina cross. This is a pretty central place in the city, so it is usual that people going by car to the stadium take the M30 road, which is a fast way to get to many places in town. Most of the points are placed in this road, so the results are coherent.

More information can be obtained. Normally, traffic in M30 is quite fluid even in the peak hours. Thus, it suitable that even with a high occupation and road load, intensity still has a higher value. In cluster number 4, the center for intensity (i.e. *intensidad*) is much lower, but still the

values for occupation and road load are high. This may imply that there are also urban points with difficulties not appearing yet. The highest value for intensity in cluster 4 is 936. Thus, a lower arbitrary values of 75% have been considered for occupation and road load. The points are shown in the map of Figure 4.29.

This time, the critical points are placed in the streets and not in M30. Surprisingly there are some of them in the south, but there are also points close to the stadium's influence.

Figure 4.29: Points with critical values of road load with matches at SB in cluster 3



4.3.2 Application to time intervals affected by matches in the Wanda Metropolitano stadium

The time intervals that are affected by a match in WM give 30,716 measures for a total of 3,998 points. The application of the Elbow method implies again that 5 is a valid value to get differentiated clusters. The distances between clusters and between the elements within a cluster are, again, separated by one order of magnitude. This means that the choice of the Elbow method is good enough. Figure 4.30 shows the centers of the clusters.

Figure 4.30: Centers for intervals affected by matches in WM

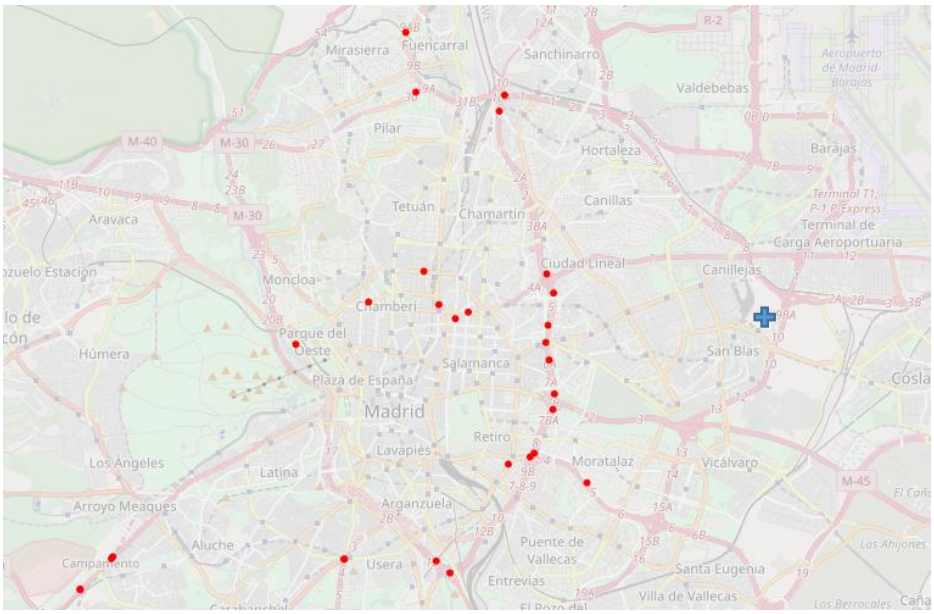
```
> kmwM2$centers
  intensidad ocupacion    carga      prec   franja
1   589.2824  7.838312 28.88583 0.8128820 6.501096
2  2273.2715 10.008365 34.43118 0.7770342 6.533080
3  3971.7662 11.273148 35.75000 0.7631944 6.430556
4   185.3890  8.142966 19.56953 0.8307423 6.484903
5  1201.8561 10.403465 34.93386 0.8307402 6.523780
```

It is a very similar result to the case of SB. However, for cluster 4, the center of the road load is more coherent, since it is the lowest value of all. The peak of intensity, occupation and road load is in the same cluster, number 3. It contains measures from 381 different. Taking into account that the minimum value of intensity is 1,738 vehicles per hour, initially it can be assumed that these points belong to M30, A-2 or branches.

Figure 4.31 shows the points with occupation and road load higher than 70%.

The blue cross is the location of the stadium. To arrive there by car, drivers must use A-2 or the roads around. The easier and more direct ways to get to them is using M30 again, which has the characteristics mentioned in section 4.2.3.1. There are still some outliers in the south and there are also some points in big streets within the city (Avenida de Filipinas, Paseo de la Castellana, Calle de María de Molina and Calle de Serrano) affected with critical values.

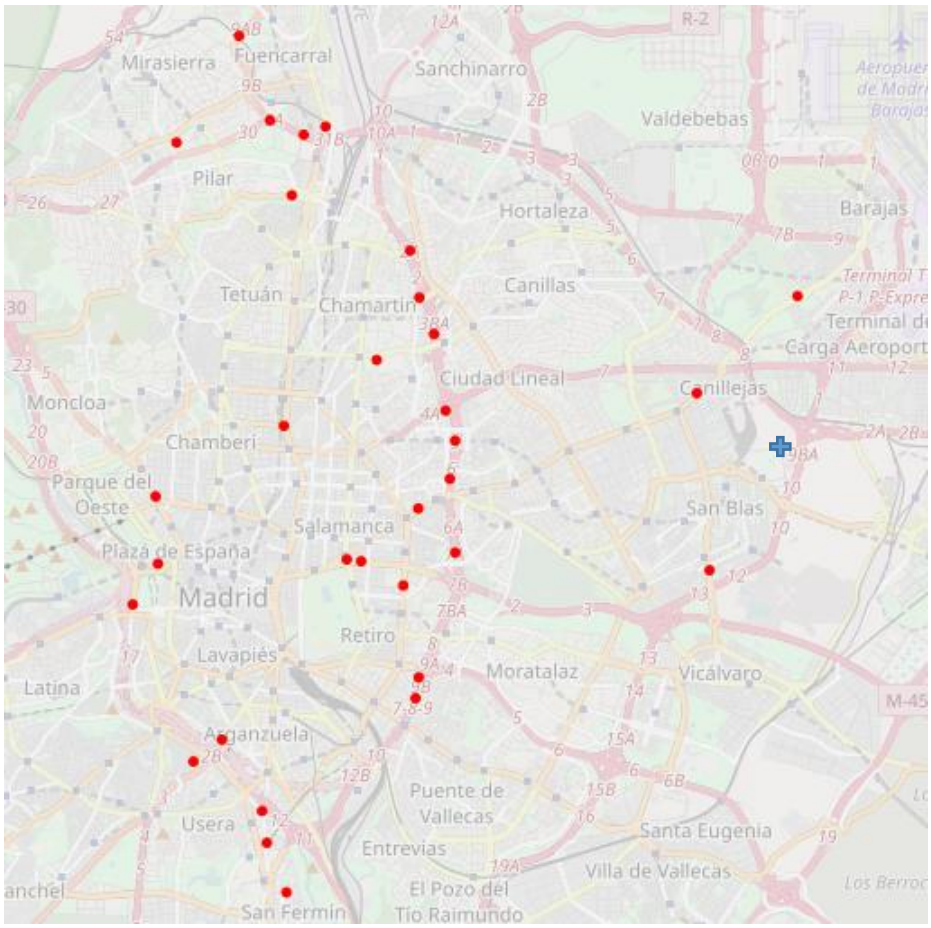
Figure 4.31: Points with critical values of road load with matches at WM in cluster 3



For cluster 5, there is a medium value of intensity in the center. However, there are also measures with critical values for road load and occupation. The critical points in this cluster may be placed in zones with smaller streets, where it is harder to have a fluid traffic. There are 1,057 points included in these measures and the ones with values greater than 80% for occupation and road load are shown in Figure 4.32.

M30 points also appear here for moments that traffic is less fluid. On the other hand, there can be seen points in districts like Salamanca and Retiro where there are more difficulties like short streets, zebra crossing and traffic lights. There are some outliers in the south and the airport as well. There is a point in Canillejas that may be influenced or not by matches at WM since it is placed in the way to the airport as well.

Figure 4.32: Points with critical values of road load with matches at WM in cluster 5



4.3.3 Application to days affected by rainfalls

As seen in previous sections, December 2017 was a pretty dry month while March 2018 was much affected by rains. In order to see how days with intense rainfalls affect to traffic congestion, the data from both months are merged to be processed together. The quantity of rainfalls are taken with a daily frequency. It would have been more precise to perform the analysis with, at least, hourly information. It is not the same having for a day a value of 20 liters per square meter in just four hours or through the whole day. Nonetheless, with the available information, the analysis and conclusions below have been taken.

The combined object has an average value of quantity of rainfalls of 2.77 liters per square meter. The median is 13.95 and the mode is 0. Given these calculations, it is considered to study data above the median value, which results in a group of 145,249 measures with values for quantity of rainfalls between 14.1 and 27.9. The Elbow method sets 5 as a good enough value for the number of clusters. Figures 4.32 and 4.33 show the centers of all the five clusters and the distribution of measures in a histogram.

Figure 4.33: Cluster centers for data affected by rainfalls

```
> kmRF$centers
  intensidad ocupacion    carga     prec   franja
1  2318.1140 12.699053 41.33598 20.41085 5.449621
2   558.3647  8.879592 29.06848 20.18065 5.326189
3   137.6828  6.064717 13.93506 19.81973 4.033441
4  4180.0282 14.046875 49.84736 20.36154 5.294471
5  1180.3032 12.423670 38.84591 20.42073 5.321214
```

As in most of previous applications, as center values for intensity increase, so does road load and occupation. It is also worth to mention that the values for quantity of rainfalls are similar, around 20 liters per square meter. The number of elements assigned to each cluster decreases as critical values for traffic congestion increases, for the same reasons explained in previous sections. Driving under the rain is more dangerous because the braking distances increase. For this reason, for cluster 1, which has the biggest value for intensity, it has been considered as critical a value of road load above 90%. Figure 4.35 shows the critical points.

Accesses to the city and, again, M30 are the most affected roads in this cluster. These roads are the ones supporting more vehicles by minute through them. Given the characteristics of the cluster, it is coherent that the critical points are found there.

Cluster 2 is taken into account next. It has a much lower value of intensity in its center, even though the value for road load is still high enough to consider that there could be critical points in it. Given the value of intensity, it is possible that the points in this cluster are mainly urban ones. Then, a value of 90% for road load and 80% for occupation are taken. Figure 4.36 shows the map with the critical points.

Figure 4.34: Elements assigned to each cluster from data affected by rainfalls

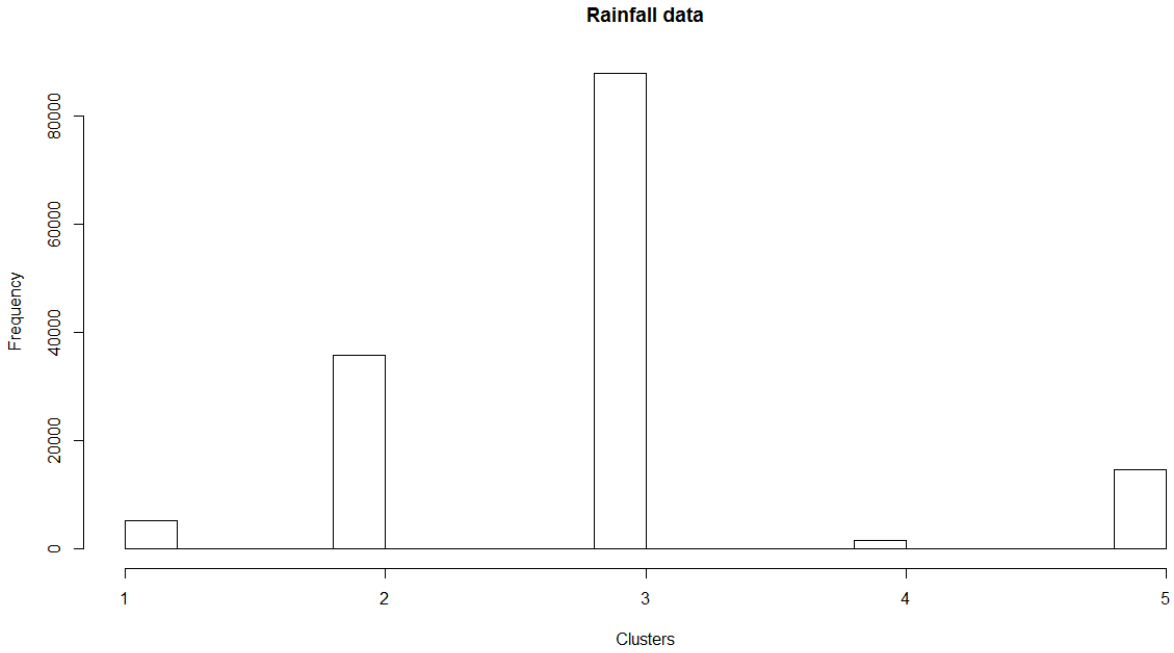


Figure 4.35: Points with critical values of road load with rainfalls for cluster 1

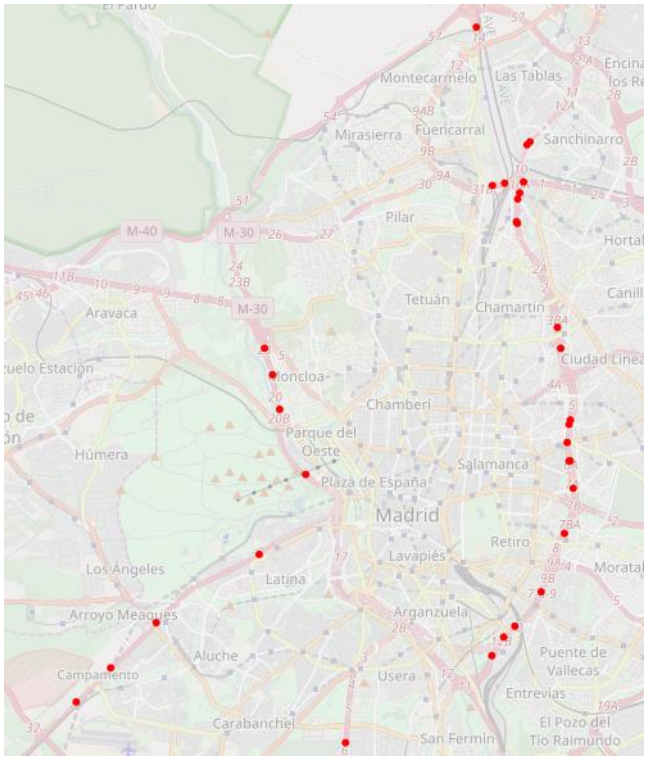
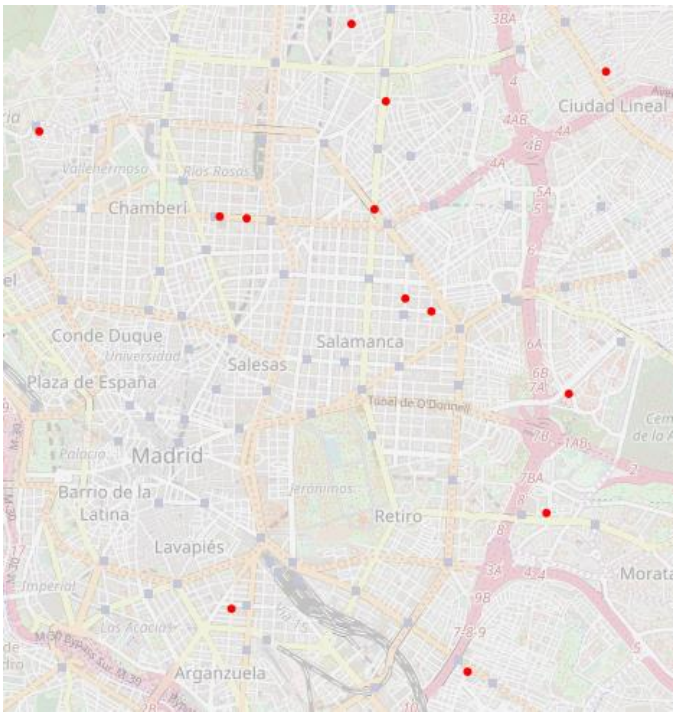


Figure 4.36: Points with critical values of road load rainfalls for cluster 2



In this map, more urban points are shown. Some secondary streets within the city like Conde de Peñálver, Ortega y Gasset, María de Molina along with main streets like Paseo de la Castellana or Francisco Silvela are affected. Here, average speed is lower so the number of vehicles per hour in one measure point is lower than in points of cluster one.

The conclusions in this section show that the previous ones from the analysis of the data in December 2017 can be roughly applied to data in days with intense rainfalls. Data are divided in clusters where the geographical location has influence, and different measure values can imply different congestion degrees. Since results are roughly similar to the mentioned case of December, it can be concluded that rainfalls does not have a distinctive impact in daily traffic.

4.4 Congestion definition

Traffic congestion can be defined in multiple ways. For instance as a situation when traffic is moving at speeds below the designed capacity of a roadway or there is an excess of demand for a road. In (Rao and Rao, 2012), the authors state that congestion is a function of a reduction in speeds, which is the direct cause of loss of time and leads to increased vehicle operating costs, fuel consumption, and emissions of air pollutants and Green House Gases (GHGs). Therefore, setting a threshold that is directly related to travel speeds seems the most appropriate approach. However, the data used in the current do not include the measure of average travel speed for urban points, which are the majority. Hence, this analysis must use an alternative way to determine if there are traffic congestions in a measure point.

The analysis in sections 4.2 and 4.3 has shown that the points affected by critical values of road load also appear in points geographically far from the places where the events are happening. K-means was applied to all the measures with time intervals affected by the succession of these events in 4.3. It showed that roads far from the event points that serve as access to the city (mainly M30) and other main streets within the city are affected as well. Thus, this analysis has served to identify possible points of conflict beforehand.

In order to close the analysis, this section is going to take the information obtained in section 4.3 and use it to determine generic parameters to establish when there is a traffic congestion issue. Table 4.1 collects the centers for the five clusters in each case study grouped by similar values.

Table 4.1 shows that cluster division has similar characteristics for the three case studies. Their centers are placed in similar ranges. This means that the following analysis would have a very similar result for the three sets of data. Here, it is performed using the data for WM.

The data and conclusions from sections 4.1, 4.2 and 4.3 suggest that intensity is not a reliable characteristic to determine if there is a traffic congestion in this analysis. It can be really high, keeping occupation and road load low, and it could also have a not too high value while the

other two parameters are high. The situation depends on the kind of road where measures are taken. Table 4.2 shows some examples of measures.

Table 4.1: Cluster centers for each case study

Bernabéu	Metropolitano	Rainfalls
Intensidad: 3778.9770 Ocupacion: 8.678571 Carga: 35.02041 Prec: 3.389541 Franja: 6.492347	Intensidad: 3971.7662 Ocupacion: 11.273148 Carga: 35.75000 Prec: 0.7631944 Franja: 6.430556	Intensidad: 4180.0282 Ocupacion: 14.046875 Carga: 49.84736 Prec: 20.36154 Franja: 5.294471
Intensidad: 2283.1548 Ocupacion: 7.474513 Carga: 25.90639 Prec: 3.443188 Franja: 6.667285	Intensidad: 2273.2715 Ocupacion: 10.008365 Carga: 34.43118 Prec: 0.7770342 Franja: 6.533080	Intensidad: 2318.1140 Ocupacion: 12.699053 Carga: 41.33598 Prec: 20.41085 Franja: 5.449621
Intensidad: 1271.8780 Ocupacion: 7.356709 Carga: 27.24443 Prec: 3.540049 Franja: 6.608431	Intensidad: 1201.8561 Ocupacion: 10.403465 Carga: 34.93386 Prec: 0.8307402 Franja: 6.523780	Intensidad: 1180.3032 Ocupacion: 12.423670 Carga: 38.84591 Prec: 20.42073 Franja: 5.321214
Intensidad: 601.9827 Ocupacion: 6.419997 Carga: 25.83498 Prec: 3.848503 Franja: 6.553042	Intensidad: 589.2824 Ocupacion: 7.838312 Carga: 28.88583 Prec: 0.8128820 Franja: 6.501096	Intensidad: 558.3647 Ocupacion: 8.879592 Carga: 29.06848 Prec: 20.18065 Franja: 5.326189
Intensidad: 183.4401 Ocupacion: 6.806907 Carga: 17.55071 Prec: 3.501070 Franja: 6.615637	Intensidad: 185.3890 Ocupacion: 8.142966 Carga: 19.56953 Prec: 0.8307423 Franja: 6.484903	Intensidad: 137.6828 Ocupacion: 6.064717 Carga: 13.93506 Prec: 19.81973 Franja: 4.033441

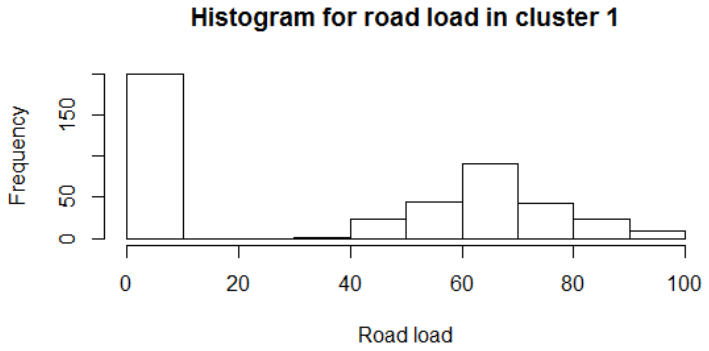
Table 4.2: Examples of measures

Id	Fecha	Tipo_elem	Intensidad	Ocupacion	Carga	franja
1017	2017-12-16	M-30	3179	5	0	7
4388	2018-03-08	URB	800	73	67	7

The measures in Table 4.2 belong to Table 4.1, the first to the cluster in the first row and the second to the cluster placed in the fourth row. These two measures show clearly the main difference between both kinds of road. M30 is taking an average of 3,179 vehicles per hour in that interval, and it does not imply congestion. It is a much higher value than the 800 in the second one, but in this case the road load rises to 67% and occupation to 73%. It would be really useful to count

with the average travel speed in urban roads, since it would give additional information about the traffic congestion grade. Unfortunately, this is only available for M30 points.

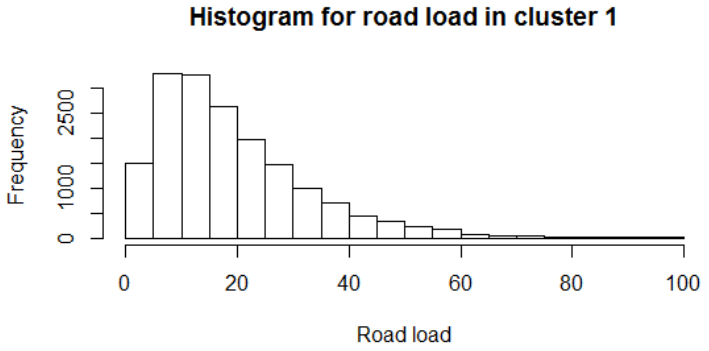
Figure 4.37: Histogram for road load values in first cluster



In the measures of the first cluster for WM in Table 4.1, the minimum value for intensity is 3,124 vehicles per hour. This is much higher than the center of the cluster placed in row 2, which has the second greater value for intensity. Even though this is a high value, the histogram of the values for road load in Figure 4.37 shows that there are both high and low values.

On the opposite side, the cluster placed in the fifth position has the lowest average value of intensity among all the clusters, as we can see in Figure 4.38. It has also high and low values for road load.

Figure 4.38: Histogram for road load values in fifth cluster



This analysis supports the decision of not considering intensity to determine if a congestion for a point of measure and a time interval exists. Moreover, and as pointed out before, measures of average speed do not exist for urban points. Thus, this work only uses road load and occupation to determine congestions. Section 3.1 showed that the road load is a percentage parameter that takes into account intensity, occupation and the capacity of the road.

Figures 4.39 and 4.40 shows the histograms for road load and occupation for data affected by matches at WM.

Figure 4.39: Histogram for road load

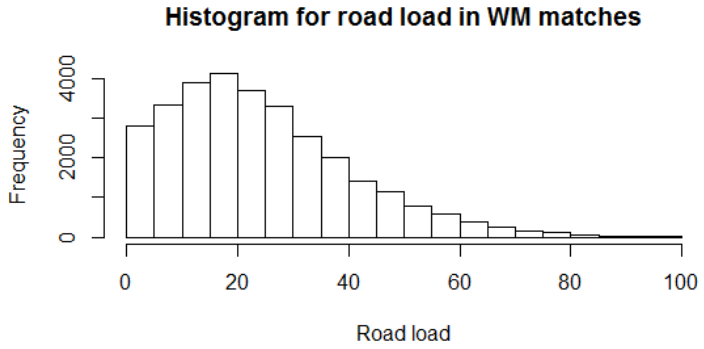
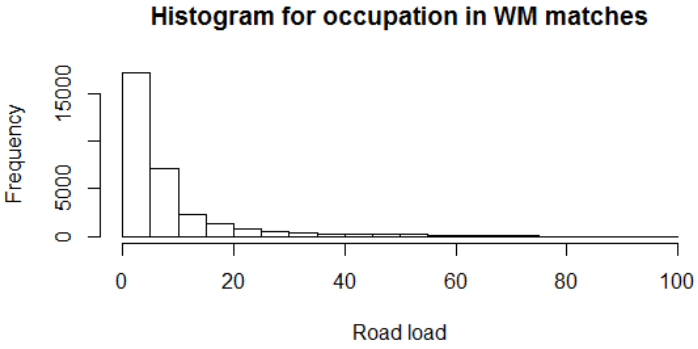


Figure 4.40: Histogram for occupation



As expected, the appearance of values decreases as they become more critical. The road load is, in broad terms, more sensitive than occupancy: only 8.92% of measures has a greater value for the second. Ideally, this information combined with the average speed could help to define

grades of congestion more precisely. However, since this feature is not taken for urban points, which constitute the majority of them, this is not possible.

This links the definition of congestion to set an arbitrary and statistically logical value that fits the analysis purposes. Setting the threshold in 60% for road load, 3.72% of the measures affected by a match in WM would be determined to have a congestion issue in the road where the point is placed. Here, it assumes that when the road load overcomes 60%, traffic is not completely fluid and it starts to be reasonable to find an alternative way to get to the desired place. This same reasoning is applied to occupation, setting its threshold value in 60% to cover the cases of that 8.92% of measures that has a greater value for this characteristic than for road load. Finally, an additional consideration is taken regarding the average travel speed. It is only available for measures in M-30 points, as said earlier. The speed limit is 70 kilometers per hour in tunnels and 90 kilometers per hour in the surface. Hence, an arbitrary value of 40 kilometers per hour is set as threshold for a congestion.

4.5 Conclusions for Chapter 4

In this Chapter, the clustering algorithm k-means has been applied to the processed data. This have been performed in three different ways:

- To the whole data set of one month
- To measures geographically close to the points where the events are happening
- To measures of all points the time intervals where the events have influence.

The results for every application are distributed in similar ways with the exceptions commented in each section. The main conclusion is that the measures are distributed in clusters with influence of the traffic congestion situation and the different types of roads that can be found in the city.

Finally, the information and conclusions extracted are used in section 4.4 to define global conditions for parameters of road load, occupation and average speed. When these are

accomplished, we can determine that there is a traffic congestion issue for that point and time given.

Chapter 5 - Short-term prediction of traffic congestion

The goals of the analysis in this chapter is being able to predict congestions in the next time interval using the data from the current one. As pointed out in the introduction, this problems is addressed here as one of classification. When discussing data processing in section 3.5, it was mentioned that two new columns are added to the objects containing the traffic data, *act* and *act_pred*. The goal of these columns is to determine if there is a traffic congestion issue in that interval for that measure point in the first case (***act*=1**, positive), and if it will happen in the one immediately next (***act_pred*=1**, positive). So, *act_pred* contains the same info as *act* but in the previous interval. The goal is to develop models able to classify points using this column, which would act as a prediction for the next time interval. This would say to a user in a hypothetical system "watch this street out, there will be congestion in the next interval". For this classification problem, decision trees and k-nn algorithms are contemplated. The variables used in the classifiers throughout the whole chapter are intensity, road load, occupation, quantity of rainfalls and activation.

5.1 Prediction with decision trees

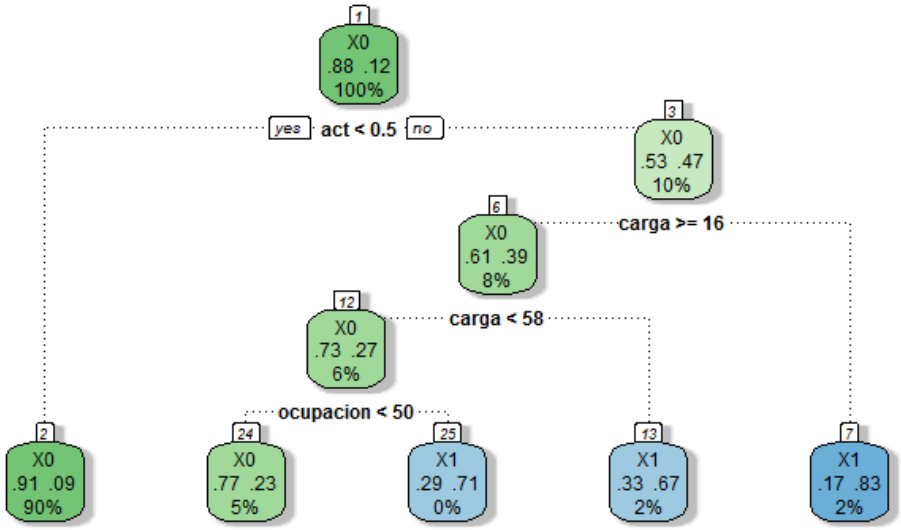
As explained in section 2.3.2.1, decision trees are a supervised learning algorithm used for classification. It learns to do it from the labeled characteristics of the data introduced as training. The function that runs the algorithm and implements all the adjacent operations is *decTreeCV.R*, which is described in the Appendix B. As this part of the work is focused on the prediction of the traffic congestion issues when certain events are happening, the data isolated in section 4.2.3are considered. These data comprehend the measures for all points in the time intervals when the considered event object has influence.

5.1.1 Prediction with Santiago Bernabéu matches

Data from points of measure with time intervals affected by matches at SB for the months of December 2017 and March 2018 are isolated as explained in section 4.3.1. These data consist of 27,767 rows. Of those, 20% have been used as testing data, while the rest have been used as training data using a 10-fold cross-validation method. Figure 5.1 shows the resulting tree. The

properties used to build it are intensity, occupation, road load, quantity of rainfalls, and activation. The target of classification is the prediction of activation (i.e. *act_pred*). Figure 5.2 shows the numeric result for this prediction.

Figure 5.1: Decision tree for predictions when there are matches at SB



The resulting tree considers the following attributes (from the root to the leaves, and from left to right): the current activation (i.e. *act*), road load (i.e. *carga*) and occupation (i.e. *ocupacion*). This allows to decide in the leaves if there will be an activation in the next time interval (i.e. *act_pred*), which is the goal of classification. Green values in leaves correspond to *act_pred* = 0, and blue ones to *act_pred* = 1.

The tree discards any measure that does not have a traffic congestion, which makes impossible to predict the ones that open a series of time intervals with congestion. Once this is settled, it stablishes values of road load and occupation to make the decision.

In the nodes we can see some parameters. In node number one, on top of the tree, “X0” is the tag assigned to the current measure when the process is found in that node. This can change if the final node is, for example, number 12. The percentage below specifies the number of measures of the testing data that arrive to that node following the process of decision.

The numeric results in Figure 5.2 show that accuracy, which is the rate of good predictions, is 90,13%. However, precision is too low. Precision is, in this case, the ratio of true positives, i.e. the ratio of the times that *act_pred*=1 was correctly predicted. Since there are very few times that a congestion exists for the next time interval (691 as the results show), it is of special importance to have the precision value as high as possible. 31,40% means that less than one out of three times that an activation is predicted, it is done correctly.

Figure 5.2: Numeric result for prediction for SB data

```
> dtreeSBCV <- decTreeCV(data_SB2, 0.8)
[1] 27767
   intensidad ocupacion carga prec act act_pred
1:       2693         9     0    0   0        0
2:       1259        11     0    0   0        1
3:       1960         8     0    0   0        0
4:       1196         8     0    0   0        0
5:       1944        10     0    0   0        0
[1] "The accuracy of the model is:  0.901314604718 17"
[1] "The precision of the model is:  0.314037626628075"
[1] "Number of activations:  691"
[1] "Number of activations detected:  217"
```

5.1.2 Prediction with Wanda Metropolitano matches

Data from points of measure with time intervals affected by matches at WM for the months of December 2017 and March 2018 are isolated as explained in section 4.3.2. These data consist of 30,716 rows. Of those, 20% have been used as testing data, while the rest have been used as training data using the 10-fold cross-validation method. The tree built is shown in Figure 5.3. The properties used to build it are intensity, occupation, road load, quantity of rainfalls, and activation. The target of classification is the prediction of activation. Figure 5.4 shows the numeric results for this prediction.

The tree has the same aspect that in the case of the SB data. Thus, the interpretation is quite the same with slightly different values in the percentages of measures in each node.

Figure 5.3: Decision tree for predictions when there are matches at WM

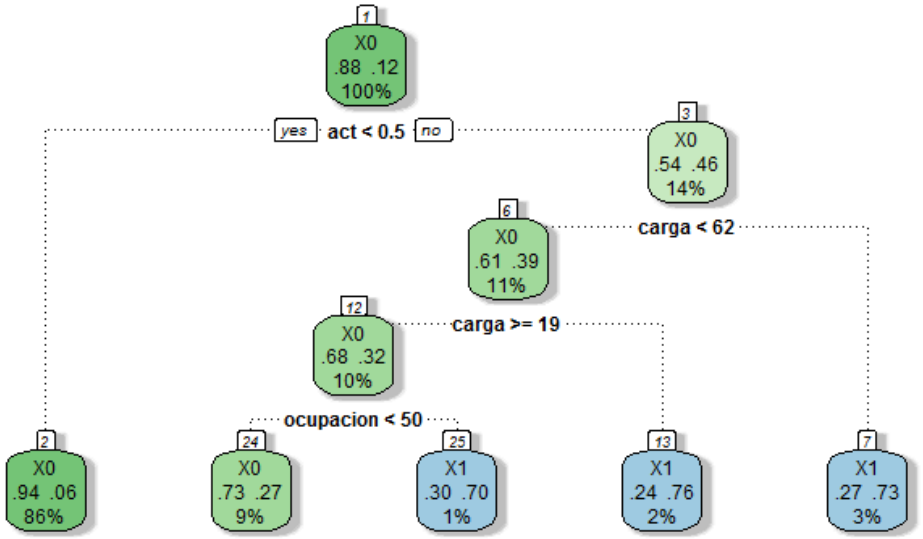


Figure 5.4: Numeric result for prediction for WM data

```
> dtreeWMCV <- decTreeCV(data_WM2, 0.8)
[1] 30716
   intensidad ocupacion carga prec act act_pred
1:       2659        10     0    0   0        0
2:       1914         6     0    0   0        0
3:       2345        18     0    0   0        0
4:       2443         7     0    0   0        0
5:       2635        10     0    0   0        0
[1] "The accuracy of the model is:  0.910778248127646"
[1] "The precision of the model is:  0.372222222222222"
[1] "Number of activations:  720"
[1] "Number of activations detected:  268"
```

The results are slightly better than for the case of SB. Accuracy rises to 91.07% and precision to 37.22%, which means than little more than one out of three of the activations predicted are correct.

5.1.3 Prediction with rains

Data affected by rainfalls have been isolated as explained in section 4.3.3. These data consists of 145,249 rows. Of those, 20% have been used as testing data, while the rest have been used as training data. The tree built is shown in Figure 5.5. The properties used to build it are

intensity, occupation, road load, quantity of rainfalls, and activation. The target of classification is the prediction of activation.

The tree follows the shape of the two previous examples. The main difference is that it considers first a threshold value of occupation, before checking if road load is bigger than zero in nodes 6 and 12. The results, as can be seen in Figure 5.6, are similar to the previous cases.

Figure 5.5: Decision tree for predictions in rainy days

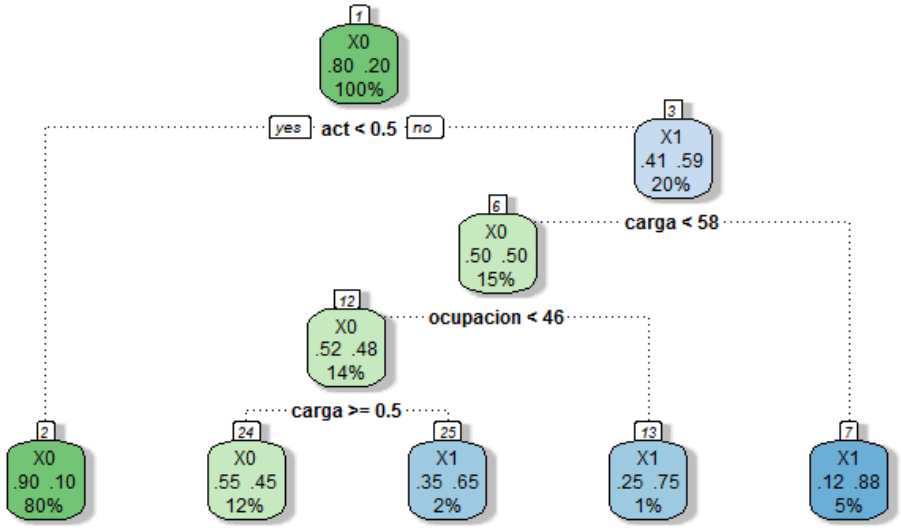


Figure 5.6: Numeric results for data affected by rains

```
> dtreeRFCV <- decTreeCV(data_RF2, 0.8)
[1] 145249
   intensidad ocupacion carga prec act act_pred
1:        458        16     0 14.1   0        0
2:        318         3     0 14.1   0        0
3:        700         3     0 14.1   0        0
4:        975         5     0 14.1   0        0
5:       1452        15     0 14.1   0        0
[1] "The accuracy of the model is:  0.850528417501463"
[1] "The precision of the model is:  0.331382337751764"
[1] "Number of activations:  5809"
[1] "Number of activations detected:  1925"
```

Accuracy is a bit lower here with a value of 85.05%. This case has the same problems as the earlier ones. Precision still has a low value for the context, 33.14%, which means one out of three true positives.

5.2 Prediction with k-nearest neighbors

As explained in section 2.3.2.2, k-nn is a supervised learning algorithm used for classification. The example to classify is assigned to the most voted class between its k nearest neighbors, being k a parameter introduced by the user. The function that runs the algorithm and implements all the adjacent operations is *kNearNeighCV.R*, which Appendix B describes. Again, data isolated in section 4.1.3 are taken to perform the predictions.

In all the analyses, data are preprocessed to work with the algorithm. 80% of rows are used for training and 20% for testing using a 10-fold cross validation method.

5.2.1 Prediction with Santiago Bernabéu matches

The data are the same used in section 5.1.1. Figure 5.7 shows the curve that plots the AUC value, against the number of neighbors, and Figure 5.8 the numeric results.

The curve and the results show that the value that maximizes accuracy is k=24 neighbors, and the algorithm choses it. Nonetheless, from approximately 11 neighbors, the results would be similar and the algorithm would take less time in training.

After the parameters, the confusion matrix is given. The columns specify the times that a 0 or 1 have been predicted and the rows specify the real value. In this case, 4,773 of the negatives predicted were correct, while 496 were really positives. On the other hand, 89 positives predicted were truly negative while 195 positives predicted were correct.

Accuracy is almost 90%, which is a good value. However, recall is 28.22%, which means that only that percentage of activations are predicted correctly, a pretty poor result given the context.

Figure 5.7: Accuracy against number of neighbors curve for SB training data

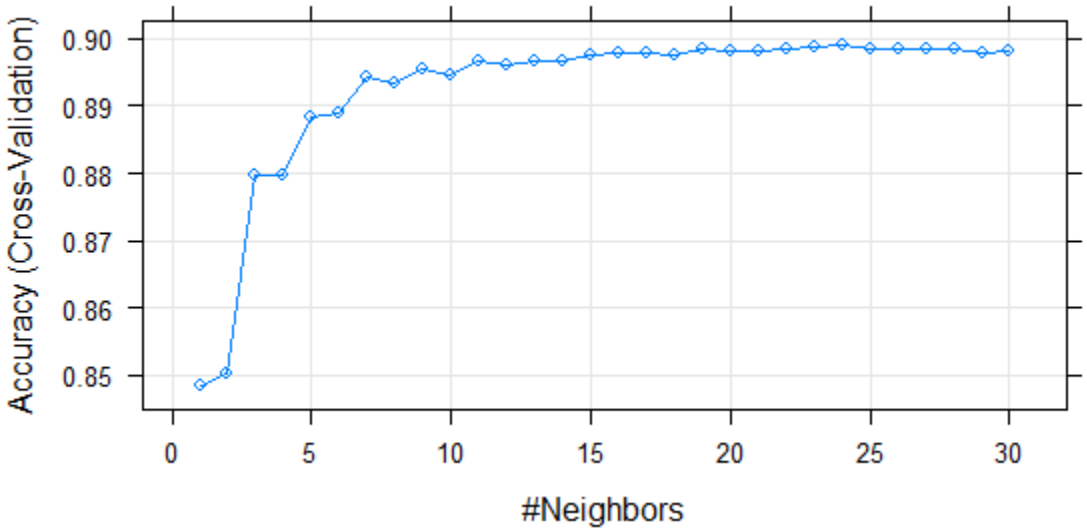


Figure 5.8: Results for prediction with 40 neighbors for SB data

```
Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 24.
[1] "Results for act_pred"
[1] "Accuracy:  0.894651539708266"
[1] "Precision:  0.686619718309859"
[1] "Recall:  0.282199710564399"
[1] "Specificity:  0.981694775812423"
    knnRes
        0    1
  X0 4773   89
  X1  496  195
```

5.2.2 Prediction with Wanda Metropolitano Matches

The data are the same used in section 5.1.2. Figure 5.9 shows the curve that plots the accuracy value, the ratio of good predictions, against the number of neighbors, and Figure 5.10 the numeric results.

Figure 5.9: Accuracy against K curve for WM training data

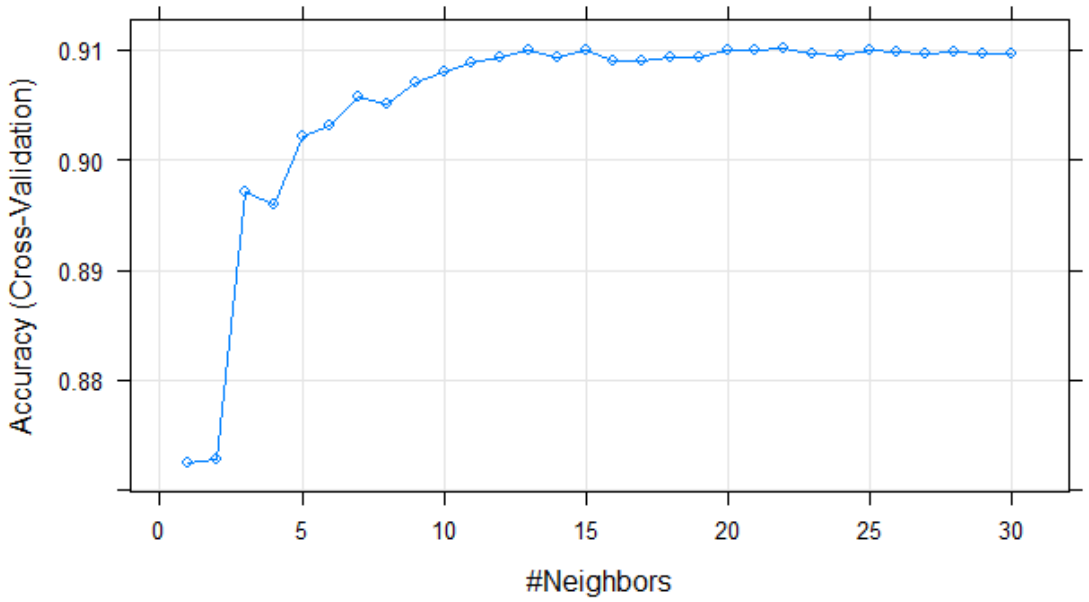


Figure 5.10: Results for prediction with 22 neighbors for WM data

```
Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 22.
[1] "Results for act_pred"
[1] "Accuracy:  0.909150113969391"
[1] "Precision:  0.708762886597938"
[1] "Recall:  0.381944444444444"
[1] "Specificity:  0.979158981925489"
    knnRes
        0    1
  0 5309  113
  1  445  275
```

Here, the value that maximizes the accuracy and the algorithm uses is 22. However, the plot shows that with values between 11 and 13 the algorithm already reaches an accuracy close to the maximum. The recall is better than the best one obtained for SB, which is a good starting point compared to the previous case. Accuracy and precision are in similar levels. Nevertheless, it is still a poor result, since only one out of three activations is predicted, and a little bit more than two out of three activations predicted are correct. Specificity is almost 98%, since there are only 113 activations predicted that turned out to be negative.

5.2.3 Prediction with rains

The data are the same used in section 5.1.2. Figure 5.11 shows a curve that plots the accuracy value, the ratio of good predictions, against the number of neighbors and Figure 5.12 the numeric results.

Figure 5.11: Accuracy against K curve for training data affected by rain

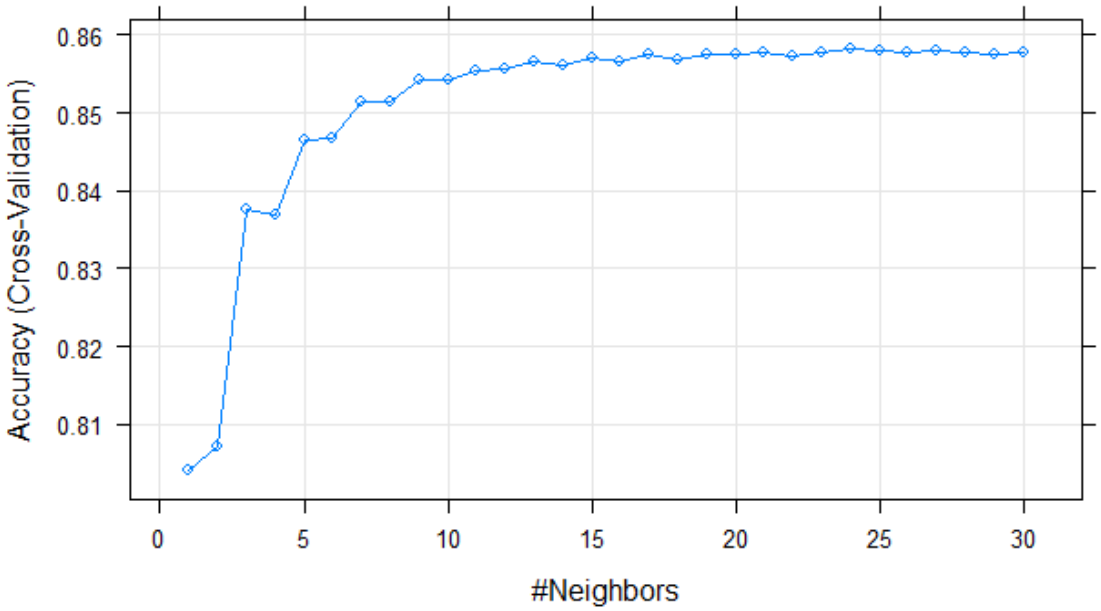


Figure 5.12: Results for prediction with 24 neighbors for data affected by rains

```
Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 24.
[1] "Results for act_pred"
[1] "Accuracy:  0.856001927777204"
[1] "Precision:  0.718548387096774"
[1] "Recall:  0.460148046135307"
[1] "Specificity:  0.954948364888124"
    knnRes
         0     1
   0 22193  1047
   1  3136  2673
```

Here, the number of neighbors is 24. The plot shows that 11 is already a good choice, since it is pretty close to the maximum. These are the best results for a prediction under section 5.2. One of the possible reasons is that they are the biggest datasets, having almost five times the number of measures of the rest. Even though accuracy and specificity have decreased lightly, results for

precision and recall are much better. With 71.85% and 46.01% respectively, these parameters indicate that almost three out of four activations predicted are correct and that almost one out of two is correctly detected.

5.3 Conclusions for Chapter 5

In this Chapter, the processed data and information and conclusions from previous ones have been used for the application of classification algorithms in order to predict in short term traffic congestions in located points.

The conditions for traffic congestion were settled in Chapter 4 for a point in a traffic interval. The target of the classification was to predict for the next time interval when there is going to be a traffic congestion issue. Decision trees and k-nearest neighbors were used with these purpose.

The results, which are fully commented in Chapter 6, are not good for predictors of these characteristics in the context given. Further actions are needed to improve the global performance of both algorithms in order to build a functional solution.

Chapter 6 - Results

In this chapter, results obtained through Chapters 4 and 5 are presented and discussed. The chapter starts discussing the division of data by geographical areas close to the critical points. Then, a cluster division using k-means with the whole set of processed data divided by months (December 2017 and March 2018) is performed in order to find patterns of classification and discern between potential critical points in terms of traffic congestion. After that, the application is performed by geographical area. Finally, the data selected are the time intervals when the events have influence for all points of measures.

The analysis started with the application of clustering to the data of December. Figure 4.2 shows its results. The distances obtained both between elements and between clusters belong to the order ten to the power of 9. However, the total distance between elements in clusters were one order below the distance between clusters, which is enough to have differentiated clusters.

This division of clusters followed a pattern that is seen in the rest of the applications with some specificities for each one. The clusters are divided by decreasing level of intensity. The other critical parameters for traffic congestion, road load and occupation, depend on the kind of road. These last parameters do not decrease as fast as intensity, which lead us to think that the clusters with higher values of intensity contain measures of fastest roads than the ones with lower values: if travel speed is higher, more vehicles can go through the road in a time interval. It can be seen as well that these critical parameters are focused in the time intervals 3 to 7. The results for March, with a greater quantity of rainfalls than those for December, were pretty similar.

Since in the previous analysis the quantity of rainfalls did not seem to have any relevant influence, the geographical division was analyzed with the data from December. Here, a similar cluster division was obtained. For SB, WM and PC, clusters with enough ratio of distance within and between clusters, and with decreasing critical parameters, are obtained. The quantity of measures assigned to each cluster increases as the critical parameters decrease. Table 6.1 presents the percentage of measure intervals affected by the event in discussion ordered by decreasing critical parameters.

Table 6.1 Percentage of intervals affected by event and cluster

Event	% Cluster 1	% Cluster 2	% Cluster 3	% Cluster 4	% Cluster 5
SB	1.49	7.08	17.88	30.45	43.12
WM	0.19	1.41	16.36	34.50	47.53

Finally, for PC, only the days after 20th of December were taken into account. Again, a similar division by clusters and quantity of measures belonging to each one.

These results do not support the initial assumption that the patterns should be somehow different, and with a higher impact of those events.

Given the results of section 4.2, in section 4.3 an analysis based directly on the intervals in which events have influence was proposed. For this analysis, the data from December 2017 and March 2018 were merged, and it considered the influence of rainfalls instead of shopping Christmas dates. The results for the three events are really similar, with bigger clusters as critical parameters decrease. Mining through data as explained in sections 4.3.1, 4.3.2 and 4.3.3 leads as well to similar conclusions to those of the previous analysis. The division is between points that still can have a great number of vehicles per hour, even though occupation and road load is high, and points where the value for intensity decreases when the other two increase. Hence, the division is between problematic points in M30 or other big avenues, problematic points in smaller streets where traffic congestion is more plausible, and non-problematic points. These statements are supported by Figures 4.29-4.36, which present the numerical results and a graphical representation of the points in a map.

Once the study of the problematic points was performed, it was possible to address building a model that predicts in the short term when it is possible to have a traffic congestion in a point. The main goal of this exercise, since the hours of the day have been divided in intervals, is to predict the issue in the previous one. This prediction problem is explored as a classification problem: given the data of the current time interval, and the category of the next interval (i.e. congestion or not, here called *activation*), is it possible to classify the current interval with that category, i.e. the current interval contains data that indicate a coming congestion? The algorithms

selected to build the model are decision trees and k-nn, which have been fully explained in section 2.1.2. Table 6.2 shows the numeric results for each algorithm and event.

First, it is worth to remember before comparing results that "precision" in decision trees is the percentage of activations detected among all activations. This corresponds to "recall" in k-nn, while "precision" in this case stands for the percentage of correct activations among all predicted.

Table 6.2 Results for predictions

Event	Decision Tree	k-Nearest Neighbors
SB	Acc: 90,13% Prec: 31,40%	Acc: 89,46% Prec:68,66% Rec: 28,22% Spec: 98,17%
WM	Acc: 91,07% Prec: 37,20%	Acc: 90,92% Prec: 70,88% Rec: 38,19% Spec: 97,91%
RF	Acc: 85,05% Prec: 33,13%	Acc: 85,60% Prec: 79,07% Rec: 46,101% Spec: 95,49%

Considering Table 6.2, both the performance of both algorithms for this problem is similar. However, as commented in sections 5.1 and 5.2, these are not good enough results to consider that the models work in a proper way. In the case of building an IoT solution that use this approach, better percentages are needed in order to make it marketable. Even though these models are not working, for example, with medical data where it is imperative getting almost 100% of good predictions since people's life depend on it, better results are expected for a working solution.

Chapter 7 - Conclusions, applications and future work

7.1 Conclusions

The main goal of this work was to provide a study of the traffic state in the city of Madrid depending on certain exceptional but recurrent events like football matches and rain. From this analysis, the work tried to develop a model for short-term prediction of traffic congestions in parts of the city. The model should be embeddable in a potential IoT solution that would acquire data and process them in real time.

The work analyzed traffic using the k-means technique to discover problematic points within the city of Madrid when those events are happening. The events considered were football matches at the stadiums SB and WM, days of intense rainfalls, and days when people go downtown such as the previous and later ones of Christmas.

The analysis identified usual congestion points under those circumstances. There are some recurrent points in almost any circumstances, like those in M30, which is one of the highways surrounding the city and is highly used. Others, previously discussed, depend on the events happening.

This analysis has three different stages that explore different hypotheses on traffic. The first stage assumes a strong dependence on event time, and the second that the more critical points when the events were happening would be the ones located close to the focal point of events (stadiums and PC). After the results got in section 4.2, these general assumptions were discarded. Then, k-means was applied using the measures of the time intervals affected by the different types of events, identifying the usual congestion points in each case. The conclusion was that these events exacerbate some usual traffic congestion points, and push to congestion some additional points depending on specific events.

Arbitrary threshold values in the parameters, as explained in sections 3.5 and 4.4, were set to state when there is a traffic congestion issue for a measure point in a concrete time interval. This

was required due to the heterogeneity of points to consider (e.g. highways vs urban places), and the data available for them (e.g. average speed only available for highways).

The last stage of this work was prediction, with the goal of identifying activations from the data in the previous time interval. This short-term prediction models would allow to forecast users about possible traffic congestion issues. This analysis used Machine Learning techniques, particularly clustering and classifiers to explore the prediction problem from the perspective of classification. models were trained for each event, getting the results in Chapter 5.

The results show that the accuracy of the algorithms predicting congestion for the next time interval was high, but they did not get good percentages taking only positive predictions. This points out the need of exploring other techniques or combine them to improve.

7.2 Application of congestion models to Smart Cities

The focus of the current work has been in the data analysis side of the problem. This is a preliminary step to design Smart Cities solution for the problem of traffic congestion, as pointed out in section 2.1. It is envisioned that the models of this work can have immediate applications, though more experimentation is needed.

The congestion analysis results from chapter 4 suggest that traffic could be diverted dynamically to other parts of the city to reduce the expected traffic jams. There are already case studies where semaphores and other information signals are managed according to the expected congestions (Oprica & Vinatoru, 2011). This congestion analysis could guide the information introduced in traffic panels, and support a finer grained control of street semaphores in identified critical areas.

A similar system is already working in cities like Madrid (Ecomovilidad, 2012). Nevertheless, there is not enough public information about it. The system is able of collecting data, which were used in this study, and these data are used for everyday traffic management. The review did not identify particular uses of this information to enable active predictions of congestion.

The traffic congestion models could inform drivers in advance of the forthcoming traffic jams and suggest alternative routes using local information. The proposed methods did not offer a ready-to-apply performance for this task. Nevertheless, the use of decision trees has the advantage of being more informative to humans than k-nn algorithm, Bayesian networks, or other statistical regression techniques. With the advent of self-driving cars, telling drivers why the car goes one direction or another, and justify such decisions may be one critical aspect.

7.2 Future Work

Based on the analysis performed and the results obtained, the following lines are proposed for future work.

This work serves as thesis to a Master's Degree in Internet of Things. As such, the model built for prediction could be improved and encapsulated to be part of an IoT solution. There are tools to build a system that use these data and perform the predictions in real time using the same sources that have been used in this work.

There are some points to consider in the complete IoT solution. The infrastructure of the sensors in the points of measure to get information in real time is already implemented. The open data Madrid's government web page, as previously mentioned, offers tools to get via Internet this information. These models could be encapsulated in a system and then build a client application where users could get notifications for conflictive points, and intervals and recommendations about traffic in real time.

In order to improve the results obtained, more data have to be used. This requires hardware resources more powerful than those available for this work, since the volume of data to process is huge. Otherwise, the processing could take such a long time that render the analysis useless for everyday routing. Moreover, additional preprocessing of data could be considered. For instance, further analysis to identity potentially dependent variables that could be removed. Of course, more techniques should be considered as well. Section 2.1 discussed that there have been works were

several Machine Learning techniques were tested in this context (van Hinsbergen et al., 2007). The conclusion of that work was that a fully working solution should work with different techniques to cover all the possible situations where one of them works better than the others.

Furthermore, Smart Cities and autonomous cars are rising parts of our lives, which are already relevant areas for research and innovation in IoT. There, systems able to predict and manage traffic congestions will be needed. What additional sensor information could make for congestion analysis and forecasting is to be explored. Also, the influence of the prediction in the behavior of the drivers. If all drivers in a congestion are advised to take another path, this may lead to new congestions in other places, giving the impression that the recommendation was wrong.

This work acknowledges the support of the following projects: “Collaborative Design for the Promotion of Well-Being in Inclusive Smart Cities"” (TIN2017-88327-R) funded by Spanish Ministry for Economy and Competitiveness; and MOSI-AGIL-CM (S2013/ICE-3019) co-funded by Madrid Government, EU Structural Funds FSE, and FEDER.

Bibliography

1. Aslam, J., Lim, S., Pan, X., & Rus, D. (2012, November). City-scale traffic estimation from a roving sensor network. In Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (pp. 141-154). ACM.
2. Bholowalia, P., & Kumar, A. (2014). EBK-means: A clustering technique based on elbow method and k-means in WSN. International Journal of Computer Applications, 105(9), 17-24.
3. Buch, N., Velastin, S. A., & Orwell, J. (2011). A review of computer vision techniques for the analysis of urban traffic. IEEE Transactions on Intelligent Transportation Systems, 12(3), 920-939.
4. Castro, P. S., Zhang, D., & Li, S. (2012, June). Urban traffic modelling and prediction using large scale taxi GPS traces. In International Conference on Pervasive Computing (pp. 57-72). Springer.
5. Chang, L. Y., & Wang, H. W. (2006). Analysis of traffic injury severity: An application of non-parametric classification tree techniques. Accident Analysis & Prevention, 38(5), 1019-1027.
6. CRAN Project (2018): https://cran.r-project.org/web/packages/data.table/index.html
7. Dia, H. (2001). An object-oriented neural network approach to short-term traffic forecasting. European Journal of Operational Research, 131(2), 253-261.
8. Djahel, S., Doolan, R., Muntean, G. M., & Murphy, J. (2015). A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches. IEEE Communications Surveys & Tutorials, 17, 125-151.
9. Ecomovilidad (2012): https://ecomovilidad.net/madrid/como-funciona-5-el-centro-de-gestion-de-la-movilidad-del-ayuntamiento-de-madrid
10. Fawcett, T. (2004). ROC graphs: Notes and practical considerations for researchers. Machine Learning, 31(1), 1-38.
11. Finkelstein, M. M., Jerrett, M., & Sears, M. R. (2004). Traffic air pollution and mortality rate advancement periods. American Journal of Epidemiology, 160(2), 173-177.

12. France, J., & Ghorbani, A. A. (2003, October). A multiagent system for optimizing urban traffic. In 2003 IEEE / WIC / ACM International Conference on Intelligent Agent Technology (IAT 2003) (p. 411-418). IEEE.
13. Hongsakham, W., Pattara-Atikom, W., & Peachavanish, R. (2008, May). Estimating road traffic congestion from cellular handoff information using cell-based neural networks and K-means clustering. In 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2008), (Vol. 1, pp. 13-16). IEEE.
14. Kodinariya, T. M., & Makwana, P. R. (2013). Review on determining number of Cluster in K-Means Clustering. International Journal of Advance Research in Computer Science and Management Studies, 1(6), 90-95.
15. Kubát, M. (2017). An introduction to machine learning (2nd ed. 2017. ed.). Cham: Springer International Publishing. doi:10.1007/978-3-319-63913-0
16. Lippi, M., Bertini, M., & Frasconi, P. (2013). Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. IEEE Transactions on Intelligent Transportation Systems, 14(2), 871-882.
17. Liu, W. M., Guan, L. P., & Yin, X. Y. (2005). Prediction of freeway incident duration based on decision tree [J]. China Journal of Highway and Transport, 1, 022.
18. Matloff, N. (2011). The art of R programming: A tour of statistical software design. No Starch Press.
19. Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.). (2013). Machine learning: An artificial intelligence approach. Springer Science & Business Media.
20. Oprica, T., & Vînătoru, M. (2011). Case study on street traffic management in intersections from the city of Craiova. ANNALS of the University of Craiova.
21. Rao, A. M., & Rao, K. R. (2012). Measuring urban traffic congestion-a review. International Journal for Traffic & Transport Engineering, 2(4), 286-305.
22. Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. In Encyclopedia of database systems (pp. 532-538). Springer, Boston, MA.
23. RStudio (2018): https://www.rstudio.com
24. Spieser, K., Treleaven, K., Zhang, R., Frazzoli, E., Morton, D., & Pavone, M. (2014). Toward a systematic approach to the design and evaluation of automated mobility-on-

demand systems: A case study in Singapore. In Road vehicle automation (pp. 229-245). Springer, Cham.

25. Stathopoulos, A., & Karlaftis, M. G. (2003). A multivariate state space approach for urban traffic flow modeling and prediction. Transportation Research Part C: Emerging Technologies, 11(2), 121-135.
26. Ayuntamiento de Madrid (2018): Tráfico: Intensidad del tráfico en tiempo real. Descripción y estructura de los datos. https://datos.madrid.es/FWProjects/egob/Catalogo/Transporte/Trafico/ficheros/PuntosMedidaTraficoMdrid.pdf
27. van Hinsbergen, J. W. C., & Sanders, F. M. (2007). Short Term Traffic Prediction Models. https://www.researchgate.net/profile/Jwc_Lint/publication/228882378_Short_Term_Traffic_Prediction_Models/links/0a85e530c5cb44961e000000/Short-Term-Traffic-Prediction-Models.pdf
28. Weisbrod, G., Vary, D., & Treyz, G. (2003). Measuring economic costs of urban traffic congestion to business. Transportation Research Record: Journal of the Transportation Research Board, (1839), 98-106.
29. Williams, B., Durvasula, P., & Brown, D. (1998). Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models. Transportation Research Record: Journal of the Transportation Research Board, (1644), 132-141.
30. Yu, H., & Kim, S. (2012). SVM tutorial—classification, regression and ranking. In Handbook of Natural computing (pp. 479-506). Springer, Berlin, Heidelberg.
31. Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of things for Smart Cities. IEEE Internet of Things journal, 1(1), 22-32.
32. Zhang, L., Liu, Q., Yang, W., Wei, N., & Dong, D. (2013). An improved k-nearest neighbor model for short-term traffic flow prediction. Procedia-Social and Behavioral Sciences, 96, 653-662.
33. Zheng, Z., Lu, P., & Tolliver, D. (2016). Decision Tree Approach to Accident Prediction for Highway–Rail Grade Crossings: Empirical Analysis. Transportation Research Record: Journal of the Transportation Research Board, (2545), 115-122.

Appendixes

Appendix A: Main functions for processing data

parseTraf.R

This function reads the information from the traffic, measure points, and weather. It makes the proper transformations and merge the info, resulting a *data.table* object.

```
parseTraf = function(dat, dprec, pmed) {

  library(data.table)
  # ------------READ DATA_TRAF---------------
  # data_traf$intensidad (Int) -> Intensity in vehicles/hour per 15 minutes.
  # data_traf$ocupacion (Int) -> Opccupation max in % per 15 minutes.
  # data_traf$carga (Int) -> % that takes into account
intensidad/ocupacion/capacity, it's a percentage per 15 mins.
  # data_traf$vmed -> Average speed for all vehicles in the measure point per
15 mins.

  data_traf <- fread(dat, sep = ";") #Using data.table to import CSV
  print("Traffic read!")
  data_traf[, hora := sub(".*[0-9]+ ", "", fecha)]
  data_traf[, fecha := as.Date(data_traf$fecha)] #In fecha only date is kept
  print("fecha and hora splitted!")

  # ------------READ DATA_CLIM---------------
  data_clim <- read.csv(dprec,
                        header = TRUE,
                        sep=";",
                        dec = ",",
                        stringsAsFactors = FALSE)
  data_clim$fecha <- as.Date(data_clim$fecha) # The type is changed so it can
be compared to data_traf
  print("Clim data added!")

  # ------------MERGE AND SHAPE--------------
  data_traf[, prec := data_clim$prec[match(data_traf$fecha,
data_clim$fecha)]] # The column with precipitations is added
  data_traf[data_traf$intensidad<=1000, color := "green"]
  data_traf[data_traf$intensidad>1000, color := "yellow"]
  data_traf[data_traf$intensidad>3000, color := "red"]
  data_traf[data_traf$hora<"04:00:00", franja := 1]
  data_traf[data_traf$hora>="04:00:00", franja := 2]
  data_traf[data_traf$hora>="07:30:00", franja := 3] # Most occupied interval
in the morning
  data_traf[data_traf$hora>="09:00:00", franja := 4]
  data_traf[data_traf$hora>="12:00:00", franja := 5]
  data_traf[data_traf$hora>="16:00:00", franja := 6]
  data_traf[data_traf$hora>="17:30:00", franja := 7] # Most occupied interval
in the evening
```

```
  data_traf[data_traf$hora>="20:00:00", franja := 8]
  data_traf[data_traf$ocupacion==-1, ocupacion := 0] #Errors in ocupacion are
set to 0
  data_traf[data_traf$intensidad>0 & data_traf$vmed==0, vmed := 50]
  data_traf[data_traf$ocupacion>=60 | data_traf$carga>=60 |
data_traf$vmed<=40, act:=1] #Issues in traffic
  data_traf[!(data_traf$ocupacion>=60 | data_traf$carga>=60 |
data_traf$vmed<=40), act:=0]
  data_traf <- data_traf[id != 0] #Rows with id=0 are dropped
  print("Traffic adapted!")

  # ------------MERGE DATA_TRAF & PMED-------
  data_pmed <- fread(pmed, sep=";", dec=",") #using data.table to import CSV
  data_traf[, x := data_pmed$x[match(data_traf$id, data_pmed$id)]] #The
column with X coordinates is added
  data_traf[, y := data_pmed$y[match(data_traf$id, data_pmed$id)]] #The
column with Y coordinates is added
  print("Traffic y Pmed merged!")

  return(data_traf)
      }
```

transformTraf.R

This function receives as input a *data.table* object from parseTraf.R and performs transformations that make it more manageable.

```
transformTraf = function(traffic){

  # ------------TRANSFORM DATA_TRAF----------
  traf_ids <- unique(traffic$id)
  traf_intervals <- unique(traffic$franja)
  traf_dates <- unique(traffic$fecha)
  n_rows <- length(traf_ids)*length(traf_dates)*length(traf_intervals)
  print(paste("Number of Rows:", n_rows))

  library(data.table)
  library(dplyr)
  library(chron)

  traffic2 <- traffic[0,]
  data_par1 <- traffic[0,]
  data_par2 <- traffic[0,]
  data_par3 <- traffic[0,]
  data_par4 <- traffic[0,]
  data_par5 <- traffic[0,]
  data_par6 <- traffic[0,]
  data_par7 <- traffic[0,]
  data_par8 <- traffic[0,]

  cont = 1
```

```
  setkey(traffic, "id", "fecha", "franja") # Keys are set for subsetting
data.table

  #The data.table is filled
  for (i in 1:length(traf_ids)){
    for (j in 1:length(traf_dates)){
      for (k in 1:length(traf_intervals)){
        print(paste("Franja:", traf_intervals[k], "Fecha:", traf_dates[j],
"Id:", traf_ids[i]))
        block <- traffic[J(traf_ids[i], traf_dates[j], traf_intervals[k]),
nomatch=0L]
        if (nrow(block)==0)
          rowsBlock <- -1
        else
          rowsBlock <- lengthBlock(block)

        #Due to inconsistency in data, comprobation of the size of block is
tested
        if (rowsBlock == nrow(block)) {
          data_row <- block[1,] #Gives format to data_row
          data_row$intensidad = round(mean(block$intensidad))
          data_row$ocupacion = round(mean(block$ocupacion))
          data_row$carga = round(mean(block$carga))
          data_row$vmed = round(mean(block$vmed))
          if(sum(block$color=="red")>=1){
            data_row$color="red"
          } else {
            if (sum(block$color=="yellow")>=sum(block$color=="green")){
              data_row$color="yellow"
            } else {
              data_row$color="green"
            }
          }
          if(any(block$act==1))
            data_row$act=1
          else
            data_row$act=0

          if (cont < round(n_rows/8)){
            l <- list(data_par1, data_row)
            data_par1 <- rbindlist(l)
          } else {
            if ((cont >= round(n_rows/8)&(cont < round(n_rows/4)))){
              l <- list(data_par2, data_row)
              data_par2 <- rbindlist(l)
            } else {
              if ((cont >= round(n_rows/4)&(cont < round(3*n_rows/8)))){
                l <- list(data_par3, data_row)
                data_par3 <- rbindlist(l)
              } else {
                if ((cont >= round(3*n_rows/8)&(cont < round(n_rows/2)))){
                  l <- list(data_par4, data_row)
                  data_par4 <- rbindlist(l)
                } else {
                  if ((cont >= round(n_rows/2)&(cont < round(5*n_rows/8)))){
                    l <- list(data_par5, data_row)
```

```
                    data_par5 <- rbindlist(l)
                  } else {
                    if ((cont >= round(5*n_rows/8)&(cont <
round(3*n_rows/4)))){
                      l <- list(data_par6, data_row)
                      data_par6 <- rbindlist(l)
                    } else {
                      if ((cont >= round(3*n_rows/4)&(cont <
round(7*n_rows/8)))){
                        l <- list(data_par7, data_row)
                        data_par7 <- rbindlist(l)
                      } else {
                        l <- list(data_par8, data_row)
                        data_par8 <- rbindlist(l)
                      }
                    }
                  }
                }

              }
            }
          }
          # l <- list(data_traf2, data_row)
          # data_traf2 <- rbindlist(l)
        }
        print(paste("Completion:", (cont/n_rows)*100, "%. Iteration:", cont,
"."))
        cont <- cont + 1
      }
    }
  }

  l <- list(data_par1, data_par2, data_par3, data_par4,
            data_par5, data_par6, data_par7, data_par8)
  traffic2 <- rbindlist(l)
  rm(data_par1, data_par2, data_par3, data_par4,
       data_par5, data_par6, data_par7, data_par8)

  traffic2[ , act_pred := shift(act, 1L, type="lead")] #Activation for
prediction
  traffic2$act_pred[nrow(traffic2)] <- 0 #Fill the value of the last row

   return(traffic2)
      }
```

minePmeds.R

This function calculates the euclidean distance of each measure point to te one passed as input and returns the ids that are below the threshold.

```
minePmeds = function(pmed, st) {
```

```
  #s. Bernabeu: x=441634 y=4478271
  #W. Metropolitano: x=449158 y=4476336

  library(data.table)
  x_SB <- 441634
  y_SB <- 4478271
  x_WM <- 449158
  y_WM <- 4476336
  x_PC <- 440121
  y_PC <- 4474614

  th_SB <- 2000 #Threshold for SB, stadium deep in the city with many pmed
around
  th_WM <- 5000 #Threshold for WM, apart from the city, less pmed around
  th_PC <- 2500 #Threshold for Plaza Callao

  pmed[, distSB := sqrt((pmed$x - x_SB)^2 + (pmed$y - y_SB)^2)] #Euclidean
distances to the stadiums
  pmed[, distWM := sqrt((pmed$x - x_WM)^2 + (pmed$y - y_WM)^2)]
  pmed[, distPC := sqrt((pmed$x - x_PC)^2 + (pmed$y - y_PC)^2)]

  ids_close <- data.table(0)
  ids_closeSB <- cbind(pmed$id[which(pmed$distSB<=th_SB)])
  ids_closeWM <- cbind(pmed$id[which(pmed$distWM<=th_WM)])
  ids_closePC <- cbind(pmed$id[which(pmed$distPC<=th_PC)])

  if (st =="SB")
    return(ids_closeSB)
  else {
    if (st == "WM")
      return(ids_closeWM)
    else
      return(ids_closePC)
  }

}
```

mineTraf.R

This function takes the list of ids close to the point selected and builds the *data.table* with the information corresponding to them.

```
mineTraf = function(traf, pmed, st) {

  library(data.table)
  # st is the Stadium of which data is required
  # minePmeds gets the ids close to the stadiums
  ids_closeSB <- minePmeds(pmed, "SB")
  ids_closeWM <- minePmeds(pmed, "WM")
  ids_closePC <- minePmeds(pmed, "PC")
  setkey(traf, "id")
```

```
  SB <- traf[J(ids_closeSB),] #Splitting of the info of the ids close to the
stadiums
  SB <- SB[, -c(18)] #Data from ATM is removed
  SB <- na.omit(SB) # Omits the rows of ids not appearing in traf

  WM <- traf[J(ids_closeWM),]
  WM <- WM[, -c(17)]
  WM <- na.omit(WM)

  PC <- traf[J(ids_closePC),]
  PC <- PC[, -c(17, 18)]
  PC <- na.omit(PC)

  if (st =="SB")
    return(SB)
  else {
    if (st == "WM")
      return(WM)
    else
      return(PC)
  }
      }
```

Appendix B: Main functions for clustering and prediction

k_means.R

This function takes as input a *data.table*, isolates the information required, performs the Elbow method and returns a kmeans object with the number of clusters selected as input.

```
k_means = function(data, nclusters){

  library(data.table)
  options(warn=0)
  library(fpc)
  library(cluster)

  data_only <- data[,-c(1:3,7:10,12,14:19)] #We select and scale the data
  # data_only <- data[,-c(1:3,7:10,12,14:16,18)] #We select and scale the
data for SB
  # data_only <- data[,-c(1:3,7:10,12,14:19)]
  # data_only <- data[,-c(1:3,7:10,12,14:16,18)] #We select the data for WM,
includes vmed
  # data_only <- data[,-c(1:3,7:10,12,14:19)]
  # data_only <- data[data$fecha>="2017-12-20",-c(1:3,7:10,12,14:17)] #We
select the data for PC
  print(head(data_only, 5))

  wss=integer(0)
  k.max=30
  for (i in 1:k.max){
    # km <- kmeans(data_only, i, iter.max = 50)
    km <- kmeans(data_only, i, iter.max = 50, algorithm="MacQueen")
    wss[i] <- sum(km$withinss)
  }
  print(paste("Dist:", sum(km$withinss)))
  print(paste("BSS/TSS:", km$betweenss, sum(km$withinss),
km$betweenss/sum(km$withinss)))

  plot(1:k.max, wss,
       type="b", pch = 19, frame = FALSE,
       xlab="Number of clusters K",
       ylab="Total within-clusters sum of squares", main = "Rainfalls data")

  km <- kmeans(data_only, nclusters, iter.max = 200)
  print(paste("BSS/TSS:", km$betweenss, sum(km$withinss),
km$betweenss/sum(km$withinss)))

  return(km)
      }
```

decTreeCV.R

This function takes a *data.table* as input, isolates the needed info, performs 10-fold cross-validation, trains a tree which is returned as output, makes the prediction and calculates the values needed to assess the performance.

```
decTreeCV = function(dat, t){

  library(data.table)
  library(caret)
  library(ROCR)
  library(rattle)

  data_only <- dat[,-c(1:3,7:10,12:15,17:18)] #We drop unnecessary data for
SB removing vmed

  print(nrow(data_only[, "act_pred"]))
  print(head(data_only, 5))
  data_only[
,act_pred:=as.factor(ifelse(data_only$act_pred==0,make.names("0"),
make.names("1")))]
  tr_set <- createDataPartition(data_only[["act_pred"]], p = t, list = FALSE)

  data_tr <- data_only[tr_set,]
  data_te <- data_only[-tr_set,]

  #----------------TREE FOR ACT_PRED-----------------
  col = "act_pred"
  rpart_ctrl <- trainControl(method = "cv",
                             classProbs = TRUE,
                             number = 10)
  dtree <- train(act_pred ~ ., data = data_tr,
                 method = "rpart",
                 trControl = rpart_ctrl)
  fancyRpartPlot(dtree$finalModel)

  #------------PREDICTION AND RESULTS--------------
  pred <- predict(dtree, data_te)

  acc <- sum(pred == data_te$act_pred)/length(pred)
  prec <- sum((pred == data_te$act_pred &
pred=="X1")/sum(data_te$act_pred=="X1"))
  print(paste("The accuracy of the model is: ", acc))
  print(paste("The precision of the model is: ", prec))
  print(paste("Number of activations: ",
              sum((data_te$act_pred=="X1"))))
  print(paste("Number of activations detected: ",
              sum(pred == data_te$act_pred & (pred=="X1"))))
  return(dtree)
      }
```

kNearNeighCV.R

This function take a *data.table* as input, isolates the needed data, performs 10-fold cross-validation, makes the prediction and calculate the performance parameters and confusion matrix.

```
kNearNeighCV = function(data_all, t){
  # data_all, data containing the complete info
  # t, percentage of rows dedicated to training
  # col, column to predict
  # k nearest neighbours

  library(data.table)
  library(class)
  library(caret)

  data_only <- data_all[,-c(1:3, 7:10, 12, 13:15, 17:18)] #We select and
scale the data
  print(head(data_only, 5))
  data_only[
,act_pred:=as.factor(ifelse(data_only$act_pred==0,make.names("0"),
make.names("1")))]

  tr_set <- createDataPartition(data_only[["act_pred"]], p = t, list = FALSE)
  data_tr <- data_only[tr_set,]
  data_te <- data_only[-tr_set,]

  knn_ctrl <- trainControl(method = "cv", number =10,
                           # summaryFunction = twoClassSummary,
                           classProbs = TRUE)
  knnPred <- train(act_pred ~ ., data = data_tr,
                method = "knn",
                preProcess = c("center", "scale"),
                # metric = "ROC",
                tuneLength = 10,
                tuneGrid = expand.grid(k = c(1:30)),
                trControl = knn_ctrl)
  print(knnPred)
  plot(knnPred)

  confM <- confMatrix(data_te, knnPred, t)
        return(knnPred)
```

confMatrix.R

This function calculates the confusion matrix of the predictions of one column of the *data.table* introduced as input and prints the performance parameters.

```
confMatrix = function(data_all, knnPred, t){
  # confusionMatrix(data, reference, positive = NULL, dnn = c("Prediction",
"Reference"), ...)
  # data_all, data containing the complete info
```

```
  # knnPred, data containing prediction for testing set
  # t, number of rows dedicated to test

  # i = nrow(data_all)-(t-1) #index where test rows start
  # confM <- table(data_all[i:nrow(data_all), act_pred], knnPred)

  # outcomes <- predict(knnPred, data_all)
  # knnRes <- ifelse(outcomes<= .5, 0, 1)
  outcomes <- predict(knnPred, data_all)
  knnRes <- ifelse(outcomes == "X0", 0, 1)

  confM <- table(data_all[,act_pred], knnRes)

  acc = (confM[1,1] +
confM[2,2])/(confM[1,1]+confM[1,2]+confM[2,1]+confM[2,2])
  pre = (confM[2,2])/(confM[2,2] + confM[1,2])
  rec = (confM[2,2])/(confM[2,2] + confM[2,1])
  spe = confM[1,1]/(confM[1,1]+confM[1,2])
  print(paste("Results for act_pred"))
  print(paste("Accuracy: ", acc))
  print(paste("Precision: ", pre))
  print(paste("Recall: ", rec))
  print(paste("Specificity: ", spe))
  print(confM)
  return(confM)
      }
```

Appendix C: Results for prediction without using cross-validation

This table contains the numerical result for the application of the classifiers selecting approximately the first 80% of rows for training and the rest for testing instead of performing 10-fold cross-validation.

Results for predictions without using cross-validation

Event	Decision Tree	k-Nearest Neighbors
SB	Acc: 70.95% Prec: 68.36%	Acc: 78.73% Prec:70.25% Rec: 20.33% Spec: 97.27%
WM	Acc: 82.26% Prec: 57.02%	Acc: 86.53% Prec: 56.88% Rec: 28.03% Spec: 96.41%
RF	Acc: 77.88% Prec: 63.59%	Acc: 79.61% Prec: 79.07% Rec: 46.68% Spec: 94.44%